Universidade Estadual de Campinas
Faculdade de Engenharia Elétrica e de Computação
Departamento de Semicondutores, Instrumentação e Fotônica

PROTOTIPAGEM DE CIRCUITOS ELETRÔNICOS EM TEMPO REAL, VIA INTERNET, COM APLICAÇÕES NO ENSINO DE ELETRÔNICA

Leandro Ferrari Crocomo

Dissertação submetida à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas, como requisito parcial à obtenção do título de Mestre em Engenharia Elétrica, sob orientação do Prof. Dr. Carlos Alberto dos Reis Filho

Banca Examinadora:

Prof. Dr. Carlos Alberto dos Reis Filho – FEEC/UNICAMP
Prof. Dr. Mauro Sérgio Miskulin – FEEC/UNICAMP
Prof. Dr. Francisco Javier Ramirez Fernandez – LME/USP

Campinas, 18 de junho de 2003

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

Crocomo, Leandro Ferrari

C872p Prototipagem de circuitos eletrônicos em tempo real, via Internet, com aplicações no ensino de eletrônica / Leandro Ferrari Crocomo.--Campinas, SP: [s.n.], 2003.

Orientadores: Carlos Alberto dos Reis Filho

Dissertação (mestrado) - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Ensino a distância. 2. Laboratórios de engenharia. 3. Laboratórios experimentais. I. Reis Filho, Carlos Alberto dos. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

Resumo

Esse trabalho propõe o desenvolvimento de um laboratório remoto voltado ao ensino de eletrônica à distância, composto de hardware e software, capaz de permitir a montagem real de uma gama de circuitos, suficientes para atender às necessidades dos cursos iniciais de eletrônica. Nesse sistema, uma matriz de interconexão controlada por software permite a conexão física entre diferentes componentes e equipamentos, de forma a constituir circuitos completos. Dessa maneira, é possível ao estudante conseguir resultados experimentais de forma eficiente, o que deve contribuir para a sua percepção sobre os fundamentos teóricos e os resultados conseguidos através de simulação.

Abstract

This work proposes the development of an educational remotely controlled laboratory composed by hardware and software, able to allow the real assembly of the electronic circuits commonly studied in the initial courses in electronics. In that system, a software-controlled matrix enables the connection between a sort of electronic components and measure or excitation equipment, in such a way that complete circuits can be wired up. This makes it possible to the student to get experimental results in a more efficient form, what should contribute to a better perception of the theoretical fundamentals of electronics and to a more critical analysis of simulation results.

"I found no basis prepared; no model to copy.
Mine is the first step and therefore a small one.
You, my readers, will acknowledge what I have achieved
and pardon what I have left for others to accomplish."
– Aristotle (284-322 B.C.)

"Things should be made as simple as possible,
but not any simpler."
– Albert Einstein (1879 - 1955)

"You cannot teach a man anything;
you can only help him to find it within himself."
– Galileo Galilei (1564 – 1642)

Aos meus familiares,

em especial, aos meus pais, irmã e avós.

Agradecimentos

Ao meu orientador pela oportunidade e confiança.

A Luciana Ribeiro pelo carinho e incentivo nos instantes de dificuldade.

Aos amigos de faculdade, entre eles, Álvaro Medeiros, Hermano Barros, José Cândido, Luiz Bonani e Paulo Person pela constante ajuda e companheirismo.

Ao Instituto de Pesquisas Eldorado pelo suporte e apoio fornecidos para a conclusão desse projeto. Em particular a Gabriela Castellano, Marcos Salenko, Sílvia Lopes, Sandra Minari e Tiago Agostini pela ótima convivência.

Aos colegas da Motorola de Jaguariúna, Hamilton Ferreira, Jozué Vieira, Renato Arradi e, em especial, a Matheus Rodrigues, pelo apoio financeiro indispensável à conclusão desse trabalho.

Aos amigos do Laboratório de Pesquisas Magnetti Marelli pela constante companhia, pelas valiosas idéias e sugestões e, acima de tudo, pela amizade estabelecida. Entre eles: Ana Flávia, André Couto, André Fortunato, Beto, Danilo Fanton, Donato, Dulciane, Fábio Lacerda, Fernando Castaldo, João Paulo, Jorge Polar, Marcelo, Marco Túlio, Marcos Pelícia, Matthieu, Murilo, Paulo Dal Fabro, Paulo Gustavo, Paulo Rodrigues, Roberto Cyrulnik e Wilson Jr.

Índice Geral

Índice de Figuras

Índice de Tabelas

Índice de Códigos

Capítulo 1 - Introdução

Dentre os incontáveis benefícios resultantes do crescente aperfeiçoamento dos computadores pessoais, da expansão da televisão e do desenvolvimento da Internet, destaca-se a educação à distância. Graças a estas ferramentas, a divulgação de conhecimentos tem se tornado cada vez mais fácil.

No caso particular da Internet, a forma predominante para esta divulgação de conhecimentos tem sido a disponibilização de textos. Diferente, no entanto, do que ocorre na leitura de um texto impresso em papel, a Internet propicia uma leitura hierárquica através dos chamados *hyperlinks.* Esta nova forma de divulgação de um texto oferece a possibilidade de selecionar o grau de detalhamento da informação tanto na sua geração como na leitura. Não basta, entretanto, somente o acesso à informação. É condição *sine-qua-non* à eficiência do processo de aprendizado a interatividade. Por isso mesmo, mecanismos de realimentação do aprendizado têm sido introduzidos, dotando de características cada vez mais dinâmicas os conteúdos das informações divulgadas.

Há ramos da atividade humana nos quais parte do aprendizado requer o exercício de tarefas experimentais. É o caso da Engenharia, por exemplo. Como, nestes casos, pode o ensino à distância ser implementado?

Como resposta natural à essa questão são freqüentes na literatura, descrições de ambientes que permitem desde a utilização remota de simuladores [1] até aqueles onde a caracterização de alguns componentes semicondutores é viabilizada [3].

Alguns dos ambientes propostos encontram aplicação restrita à área de controle e automação, permitindo ao estudante interagir com e/ou observar alguns dos experimentos clássicos da teoria de controle, como por exemplo, aqueles ligados ao pêndulo simples, duplo e invertido [4]. Nesses ambientes, a realimentação visual assume papel vital no processo de interação do estudante com o laboratório.

Um exemplo bastante interessante de educação à distância voltada para a parte experimental é encontrado em [5]. O trabalho propõe um ambiente onde o aluno, através de um aplicativo desenvolvido em JAVA, é convidado a montar alguns circuitos eletrônicos utilizando para isso uma matriz de contatos e alguns componentes e equipamentos, tais como multímetros,

fontes de alimentação e osciloscópios. Todos esses elementos são apresentados de forma gráfica e tentam reproduzir da maneira mais próxima possível o ambiente que o aluno encontraria em um laboratório real, incluindo os defeitos e problemas que esses elementos costumam apresentar. Pontos danificados na matriz de contato, componentes defeituosos, entre outros, são alguns dos problemas aos quais os alunos são expostos. Dessa forma, o autor afirma que o tempo gasto nas sessões de laboratório é sensivelmente diminuído, levando-se em conta a maior habilidade adquirida pelos alunos na identificação dos principais defeitos que uma montagem prática pode apresentar. Embora os conhecimentos propostos pelos autores sejam de utilidade eminentemente experimental, em todo o ambiente proposto não há em nenhum momento controle ou interação por parte do aluno com equipamentos ou componentes reais, considerando que o sistema é formado por um aplicativo executado localmente na máquina do usuário final.

A Universidade de Ciência e Tecnologia de Trondheim, na Noruega, através do grupo de pesquisas liderado pelo professor Fjeldly têm apresentado soluções bastante interessantes acerca da utilização de recursos remotos nas disciplinas de caracterização de componentes semicondutores, conforme referências [1], [6] e [7]. Essencialmente, todos os trabalhos descrevem os avanços alcançados na implementação de um ambiente, que possibilita aos seus usuários a caracterização de componentes e estruturas contidas em um circuito integrado de teste. À esse circuito integrado são conectados equipamentos de estímulo e de medidas através de uma matriz de interconexão (HP34970A). A interação com o usuário é realizada através da *homepage* do laboratório, e está limitada à escolha de alguma das topologias de teste pré-definidas e da configuração das excursões permitidas à alguns sinais como, por exemplo, tensões de polarização. Várias tecnologias para a implementação de laboratórios remotos são discutidas) e apresentadas como possíveis soluções para os problemas comumente enfrentados tanto na implementação da parte do sistema voltada para o cliente como para aquela alojada no servidor. Dentre essas tecnologias podem ser citadas, por exemplo, (*.NET, Component Object Model (COM+), Active Server Pages (ASP), Intenet Server Application Programming Interface (ISAPI), LabVIEW6i, eXtensible Markup Language (XML)* e *Scalable Vector Graphics (SVG).*

Iniciativas nacionais de implementação de ambientes como o descrito anteriormente são raras. O Laboratório de Sensores Integráveis e Microsistemas, da Universidade de São Paulo, vem trabalhando desde 1999, em um laboratório virtual que permite, de forma remota, que as

curvas características de um transistor bipolar sejam levantadas. No sistema desenvolvido um microcomputador equipado com uma placa de aquisição de dados e a plataforma de desenvolvimento LabVIEW 5.1 são adotados como solução [2].

A área da experimentação remota tem recebido grandes volumes de valiosas contribuições nos últimos anos, refletindo de forma bastante nítida a mudança de comportamento introduzida pela cada vez maior popularização das técnicas voltadas à interconexão e disponibilização de recursos em rede. Também vêm desempenhando relevante papel nesse cenário, as recentes evoluções que atingiram a instrumentação ligada a computadores pessoais na última década. Dessa forma, a abordagem de todos os trabalhos que nessa linha vêm sendo registrados pela literatura torna-se tarefa difícil. Dentre as inúmeras referências encontradas durante as revisões bibliográficas que serviram de base para esse trabalho, aquelas que maior influência exerceram encontram-se listadas de [2] à [14].

As abordagens ligadas à experimentação remota voltada para o ensino de eletrônica encontradas até o momento de escrita deste trabalho, oferecem sem dúvida, importantes oportunidades de aprendizado aos seus usuários. Ao mesmo tempo, entretanto, privam-no da oportunidade de interagir com o circuito sob teste, tanto em sua topologia quanto na definição dos pontos de observação dos resultados. Permitir ao usuário que defina a topologia do circuito a ser testado, juntamente com os pontos de observação dos resultados, por exigir maior consciência a respeito do comportamento esperado, influencia diretamente a eficiência do processo de aprendizado.

Respeitando essas condições, o presente trabalho discute a implementação de um ambiente remotamente controlado que permita a seus usuários a montagem de uma gama de circuitos eletrônicos suficientemente variada, para que esses possam se valer desse sistema como uma ferramenta de auxílio no aprendizado de conceitos relacionados às disciplinas da área de eletrônica.

Dessa forma esse trabalho se encaixa no cenário da experimentação remota como uma ferramenta de prototipação de circuitos eletrônicos com aplicações no ensino de eletrônica, que apresenta características complementares aos trabalhos até então encontrados na literatura. Dentre essas características, merecem atenção especial o fato de serem disponibilizados aos usuários apenas componentes eletrônicos descompromissados e instrumentos de medição e

estímulo; sendo atribuição do estudante, a definição da forma como esses componentes serão interconectados e quais estímulos a eles serão aplicados. Cabe ao usuário também a configuração dos equipamentos de medida em busca da melhor visualização dos resultados.

Outra característica relevante do sistema descrito diz respeito aos pré-requisitos necessários para sua utilização; limitados a uma conexão de baixa velocidade com a Internet e a um navegador padrão, não havendo restrições quanto a sistemas operacionais utilizados, versões de navegadores ou necessidade de instalação de *plug-ins*.

O presente trabalho divide-se em 6 capítulos. No capítulo 2, a estrutura geral do sistema é discutida e todos os aspectos relacionados à parte de software apresentados em detalhe. No capítulo 3 considerações sobre o desenvolvimento da matriz de interconexão são endereçadas, juntamente com as questões relacionadas à sua programação e roteamento. No capítulo 4 alguns exemplos de utilização do sistema são trabalhados, com seus resultados e avaliações. Sugestões de trabalhos futuros que tendem a complementar o presente são endereçados no capítulo 5. Por fim, no capítulo 6 são apresentadas as conclusões à respeito dessa nova ferramenta auxiliar para o ensino de eletrônica.

Capítulo 2 - Aspectos de Software

Nesse capítulo são apresentados aspectos relacionados com a arquitetura do sistema e as características do software que o constituem. Tais aspectos como a estrutura geral do sistema, o ambiente de desenvolvimento adotado, a base de dados implementada, os serviços de Internet, além do interfaceamento com os instrumentos de estímulo e medição.

2.1. Estrutura Geral e Funcionalidades do Sistema

A exposição em detalhe dos aspectos relativos às implementações de hardware e software pressupõe algum conhecimento do sistema como um todo; além de seus elementos e suas funções individuais também a forma pela qual essas interagem para alcançar o comportamento final desejado. Dessa forma é fundamental que a figura abaixo seja considerada nos parágrafos que se seguem, onde cada um dos elementos nela presente é detalhado.

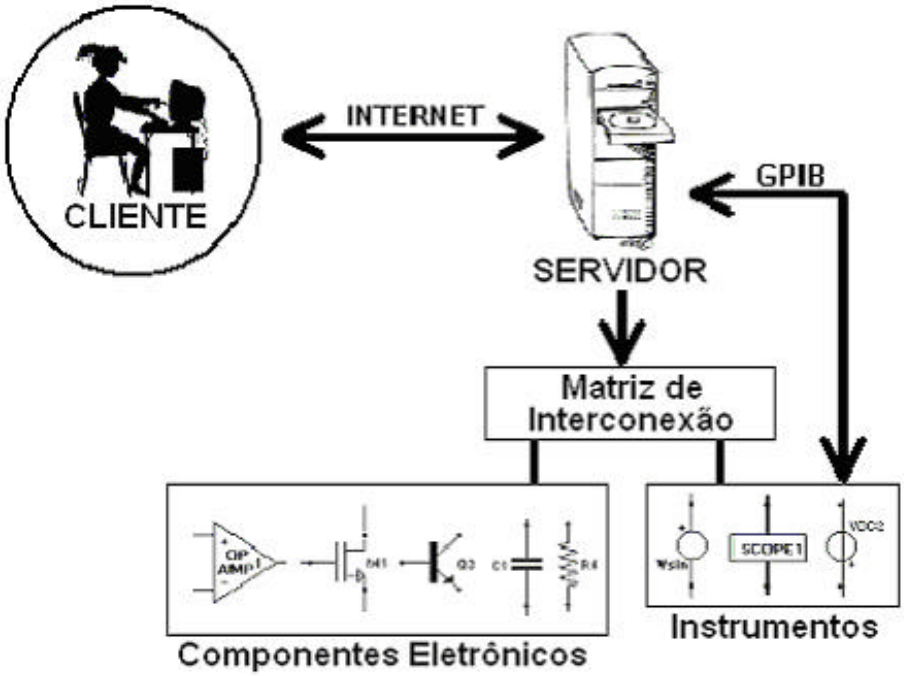


Figura 1 - Estrutura geral do sistema.

Os elementos básicos que formam o sistema encontram-se listados a seguir, juntamente com uma breve descrição de seu propósito.

- **CLIENTE**: Corresponde ao usuário do sistema. Através de um computador pessoal e uma conexão com a Internet pode enviar instruções para os equipamentos situados remotamente.
- **SERVIDOR**: É formado por um microcomputador IBM/PC responsável pelo recebimento de instruções provenientes do usuário, pela comunicação com os equipamentos, através de uma interface GPIB, e pela comunicação com a matriz de interconexão, através de sua porta paralela. Também fornece respostas que realimentam o usuário em seu ciclo de utilização do sistema. A troca de informações entre o usuário e o servidor é realizada exclusivamente através do navegador do usuário.
- **INSTRUMENTOS**: Compostos por equipamentos de medição e estímulo, tais como canais de osciloscópio, multímetros, amperímetros, fontes de polarização e geradores de sinal. Representam os elementos através dos quais o usuário interage com o circuito enviado e montado.
- **MATRIZ DE INTERCONEXÃO**: Constitui o elemento de hardware principal do sistema, capaz de prover as necessárias interconexões físicas entre os terminais dos componentes disponíveis. O detalhamento de seu projeto e construção é assunto tratado no capítulo 3.
- **COMPONENTES ELETRÔNICOS**: Representam os componentes disponíveis ao usuário em um dado momento, e consistem normalmente de resistores, capacitores, transistores, etc...

Uma vez delineados os componentes que formam o sistema, seu funcionamento global passa a ser discutido. O processo de experimentação remota com circuitos eletrônicos tem seu início do lado do cliente, que como primeira etapa deve produzir o *netlist* de seu circuito. Esse arquivo, contendo a descrição topológica do circuito, pode ser gerado através de ferramentas específicas para a captura de esquemáticos, ou então manualmente pelo próprio usuário. O

estabelecimento manual dessa descrição não deve ser encarado como tarefa penosa, visto que o número de nós normalmente considerado é manualmente administrável. Além disso, certa familiaridade com esse tipo de descrição é característica intrínseca aos currículos de engenharia em geral.

A segunda etapa do processo de experimentação remota consiste no acesso eletrônico ao laboratório. Nas páginas que formam o *site* do laboratório o usuário deve proceder, caso já não o tenha feito em ocasiões anteriores, com seu cadastro junto ao sistema; o que lhe renderá um nome de usuário e uma senha para a submissão de seus circuitos. Uma vez cadastrado, todas as submissões realizadas por um determinado usuário são registradas e resultam em importante material para realimentação acerca da utilização dos recursos do laboratório, conforme detalhado em seções seguintes.

Através de seu nome de usuário e sua senha, a submissão da descrição topológica do circuito pode ser realizada. Do lado do servidor essa descrição é recebida e analisada quanto a possíveis fontes de erro, tais como a utilização de componentes não disponíveis. Caso a descrição recebida tenha sido aprovada, a matriz de interconexão é acionada e os contatos elétricos entre os terminais dos componentes utilizados pelo usuário em sua descrição são estabelecidos. Nesse instante o circuito que até então se resumia a uma descrição abstrata passa a existir de fato.

Como última etapa desse processo, uma página contendo um formulário que permite ao usuário interagir com os instrumentos de medição e estímulo é dinamicamente produzida. A partir desse instante o usuário passa a configurar os instrumentos disponíveis e a receber de volta os resultados obtidos, dando início a um ciclo que pode se estender até que o usuário tenha completado seu estudo sobre o circuito enviado.

Ao final desse trabalho, alguns parágrafos são devotadas à discussão acerca de políticas de gerenciamento de utilização dos recursos do laboratório, onde questões como tempo de acesso de cada usuário e situações de múltiplo acesso são abordadas.

Um sistema de instrumentação remota pode, em geral, ser classificado em uma das três seguintes categorias:

- **Monitoração Remota**: Diz-se de um processo sendo executado em um determinado servidor e passivamente observado em um ou mais computadores, ditos clientes. Freqüentemente, esse tipo de utilização de recursos remotos pode constituir uma interessante ferramenta de ensino em aulas teóricas onde durante alguns instantes pode-se lançar mão da observação real de algum processo ou experimento onde não seja necessária a interação do usuário.
- **Controle Remoto**: Essa categoria acrescenta às capacidades dos sistemas de monitoração remota a possibilidade do usuário fornecer entradas que produzirão efeitos nas variáveis observadas.
- **Colaborativo**: Nessa categoria são encaixados os sistemas que permitem a múltiplos usuários acompanharem remotamente a evolução de um determinado processo, interagindo não apenas com as variáveis de entrada desse processo mas também com os outros usuários do sistema naquele instante.

De acordo com a classificação proposta anteriormente o sistema de instrumentação remota desenvolvido neste trabalho encaixa-se na categoria dos sistemas de controle remoto. No entanto, extensões que possam agregar ao sistema características de monitoramento remoto ou mesmo de um sistema colaborativo são possíveis.

2.2. Abordagens para uma interface remota.

A implementação de ambientes de experimentação remota como o proposto nesse trabalho deve prever mecanismos através dos quais a interação entre o usuário remoto e o hardware local possam se estabelecer. Dentre as várias soluções possíveis para o estabelecimento desse canal de comunicação entre cliente e servidor podem ser inicialmente destacadas as

soluções baseadas em softwares dedicados utilizados pelo cliente e aquelas que se baseiam diretamente na interface *web* e, portanto, acessíveis através de um navegador padrão.

A primeira solução consiste no desenvolvimento de um software específico para execução no lado do cliente. Essa abordagem favorece uma interface com o usuário mais rica e adequada, além de maior facilidade para se estabelecer atualizações em tempo-real das variáveis de saída monitoradas remotamente. Por outro lado, entretanto, é necessária a distribuição de arquivos executáveis que, no caso de alguma modificação do sistema, necessitariam de nova distribuição. Isso aumenta os custos e a complexidade de manutenção do sistema. No que diz respeito à segurança, esse enfoque pode ser positivo no sentido em que apenas usuários que possuam o software cliente podem acessar os recursos remotos. Por outro lado, a distribuição de arquivos executáveis pode representar alguma chance de usuários terem acesso ao código fonte ou a parte dele.

A segunda solução, onde um navegador padrão é utilizado como única ferramenta de acesso aos recursos remotos, oferece como vantagem imediata o fato de não ser necessária a instalação de softwares adicionais — exceto em alguns casos específicos onde pode existir a necessidade de instalação de *plug-ins* adicionais. Além disso, o fato de todo o software residir exclusivamente no lado do servidor facilita a manutenção do sistema. Outro aspecto diretamente relacionado com a facilidade e custos de manutenção do sistema diz respeito ao suporte multi-plataforma conseguido naturalmente com a utilização de navegadores para acesso ao sistema. Por outro lado, essa abordagem acrescenta alguma complexidade na apresentação dinâmica de resultados, exigindo a utilização de tecnologias tais como CGI — *Common Gateway Interface* — Java, ActiveX entre outras. Com relação à segurança essa implementação pode ser considerada bastante interessante pois nenhum código é executado no lado cliente. Entretanto, qualquer usuário munido de um navegador pode potencialmente acessar o sistema.

A tabela 1 resume os pontos discutidos nos parágrafos anteriores e deve servir como referência no decorrer do trabalho.

	Navegador	Software Dedicado
PRÓS	• Sem necessidade de instalação adicional de software. • Baixo custo de manutenção e atualização do sistema. • Segurança: Sem código fonte no lado do cliente. • Suportado por uma grande variedade de plataformas.	• Controle remoto e interatividade com o usuário mais facilmente implementada. • Maior flexibilidade na escolha de interface com o usuário. • Segurança: apenas usuários com acesso ao programa cliente podem acessar os recursos remotos.
CONTRAS	• Controle remoto mais complexo, necessitando CGI, JAVA, ActiveX, etc... • Segurança: Qualquer usuário com um navegador pode acessar o sistema.	• Requer a distribuição de arquivos executáveis. • Manutenção mais custosa e complexa, exigindo redistribuição de código. • Segurança: código-fonte pode ser exposto.

Tabela 1 - Comparação entre abordagens para comunicação entre cliente e servidor.

Os fatores apresentados anteriormente e reunidos na tabela 1 acima, foram decisivos na escolha de uma interface integrada ao navegador do cliente como sendo a abordagem corrente nesse trabalho.

No entanto, mesmo essa solução ainda oferece diferentes formas de implementação e, com vistas à escolha da mais adequada, três das mais presentes tecnologias capazes de permitir a construção de interfaces integradas aos navegadores padrão atuais foram examinadas.

Uma das tecnologias adequadas para a construção de interfaces integradas aos chamados *web-browsers* é a tecnologia JAVA™, da Sun Microsystems®. A linguagem JAVA, introduzida no final de 1995, é essencialmente orientada à objetos e de sintaxe muito semelhante à linguagem C. Dentre as principais vantagens desta linguagem devem ser citadas a característica "*Write Once, Run Anywhere*", que se apóia no fato de que um programa JAVA é compilado nos chamados *bytecodes* que são, durante a execução do programa, interpretados pela chamada "máquina virtual JAVA", essa sim, dependente de plataforma; aspectos de segurança foram mantidos em mente desde o início do desenvolvimento da linguagem, fazendo com que código possa ser descarregado e executado pela Internet sem causar qualquer tipo de prejuízo ao cliente. A linguagem JAVA também oferece facilidade no desenvolvimento de soluções que exijam integração em rede [15].

Outra possibilidade para o estabelecimento de mecanismos de interação remotos entre o usuário do lado cliente e os equipamentos do lado do servidor é o emprego da tecnologia ActiveX™, da Microsoft®. ActiveX deve ser compreendido, na verdade, como um conjunto de tecnologias que permitem que diferentes componentes de software interajam em um ambiente de rede, independentemente da linguagem onde foram originalmente desenvolvidos. Dessa forma, é possível ao navegador do cliente empregar um controle ActiveX que estabeleça os elementos necessários para a troca de informações com o hardware remoto.

Por fim, a última possibilidade investigada foi o emprego da chamada *Commom Gateway Interface*, ou CGI, que embora seja uma das mais antigas tecnologias para *web*, ainda é largamente utilizada para a confecção de páginas com conteúdo dinâmico. De uma forma bastante geral, CGI foi o primeiro padrão para a *web* a permitir que um usuário através de seu *navegador* realize chamadas a rotinas e programas externos residentes no servidor de uma forma independente da plataforma utilizada pelo cliente. Além dessa capacidade de permitir à servidores HTTP chamadas à código externo, CGI proporciona um mecanismo de passagem de parâmetros do servidor para esses programas e também de passagem da saída produzida por essas chamadas de volta para o navegador. Dessa forma é possível que páginas com conteúdo estreitamente ligado às entradas do usuário sejam dinamicamente produzidas.

A especificação do padrão CGI não determina a linguagem que pode ser utilizada para os programas externamente acionados pelo servidor. Na verdade, o padrão apenas define a *interface* entre o servidor HTTP e o código externo.

Uma transação entre o navegador do cliente e o servidor envolvendo chamadas a código externo utilizando CGI é composta de três fases principais, descritas abaixo e ilustradas através da figura 2.

1. O navegador do cliente envia ao servidor um pedido, incluindo o nome e a localização da aplicação que deve ser acionada, juntamente com todos os parâmetros de entrada esperados pela aplicação.
2. O servidor HTTP chama a aplicação externa passando à ela todos os parâmetros recebidos do navegador, juntamente com uma série de informações adicionais denominadas variáveis de ambiente e que contêm informações tais como endereço IP do cliente, o navegador utilizado,

entre outras.

3. Quando a aplicação externa termina sua execução ela fornece suas saídas para o servidor HTTP que as repassa ao navegador do cliente. Normalmente a aplicação produz como saída código HTML que é então enviado para o navegador que o interpreta de forma tradicional, não sendo possível ao navegador distinguir se o código recebido é estático ou se foi dinamicamente produzido por uma aplicação CGI.

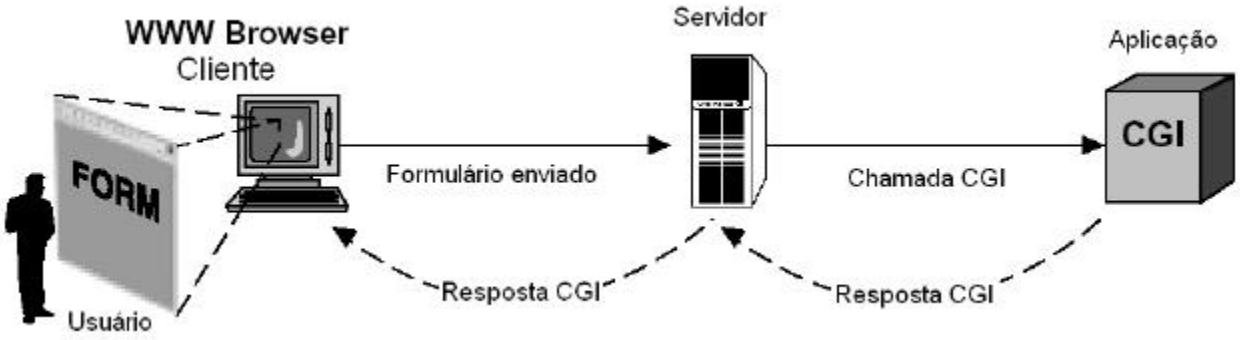


Figura 2 - Ilustração de uma transação envolvendo CGI.

A tabela 2, apresentada abaixo, resume as principais características de cada uma das três tecnologias comentadas nos parágrafos anteriores.

	CGI	JAVA	ActiveX
Capacidade de interagir com controles tais como chaves, knobs, etc...?	Possível através de imagens mapeadas, porém limitado.	Sim.	Sim.
Permite ao cliente fornecer entradas na forma de texto?	Sim, através de formulários HTML.	Sim, entretanto de forma mais complexa que CGI.	Sim, entretanto de forma mais complexa que CGI.
Suporte para várias plataformas (Windows, UNIX, Linux, etc...) ?	Sim, todos os navegadores trabalham com CGI.	Sim, a grande maioria dos navegador aceita código em JAVA.	Não. Apenas o Internet Explorer, rodando em Windows, aceita controles ActiveX.
Nível de segurança?	Possibilidade de dano ao servidor em sistemas mal projetados. Nenhum risco ao cliente.	Poucas possibilidades de problema, pela própria construção da linguagem.	Existe a possibilidade de danos ao cliente.

Tabela 2 - Comparação entre JAVA, ActiveX e CGI.

A consideração dos argumentos expostos nos parágrafos anteriores, juntamente com os seguintes aspectos:

- necessidade de se interagir com equipamentos através de diferentes interfaces (porta paralela para programação da matriz de interconexão e GPIB para controle dos instrumentos);
- necessidade de oferecer uma solução acessível a qualquer cliente, utilizando diferentes sistemas operacionais ou plataformas de hardware;
- não deve ser necessário nenhum recurso além de um navegador padrão e uma conexão discada com a Internet;

foram pontos decisivos na adoção da *Common Gateway Interface* como a solução para o estabelecimento da interface com o usuário final.

Todo o software que integra esse trabalho foi desenvolvido utilizando a linguagem de programação G, em ambiente LabVIEW 6.1. Antes da apresentação e análise de alguns trechos de código ou mais apropriadamente diagramas que compõem o sistema, uma breve introdução à linguagem adotada e seu ambiente de desenvolvimento é apresentada.

2.3. A Plataforma de Desenvolvimento

O ambiente de desenvolvimento de software LabVIEW™, — *Laboratory Virtual Instrument Engineering Workbench* — produzido pela National Instruments® teve sua origem em abril de 1983, e desde então, através de uma contínua evolução, um número cada vez maior de recursos, além da compatibilidade com diferentes sistemas operacionais, vêm sendo agregados. Atualmente o ambiente encontra-se em sua versão 6.1.

Figura 3 - Tela de abertura LabVIEW 6.1

LabVIEW é um ambiente de desenvolvimento de software genérico tal como tantos outros, como, por exemplo, Borland C++ Builder™, Visual Basic™, Kylix™, etc... No entanto, o

ambiente LabVIEW difere daqueles citados em um aspecto bastante importante: A grande maioria dos ambientes de programação utiliza linguagens baseadas em texto para criar linhas de código, enquanto no ambiente LabVIEW uma linguagem gráfica, denominada linguagem G, é utilizada para criar programas representados por diagramas de blocos [16].

Um programa escrito em linguagem G, ou por simplicidade, escrito em LabVIEW, é tradicionalmente denominado um instrumento virtual, ou VI — *Virtual Instrument* — e possui dois elementos fundamentais a saber: um painel frontal responsável por abrigar a interface com o usuário, composta normalmente de controles e indicadores nas mais variadas formas, e um diagrama onde a funcionalidade do VI é graficamente definida. Cada elemento presente no painel frontal possui um terminal correspondente no diagrama, através do qual informações podem ser lidas desse ou escritas para esse elemento, seja ele um controle ou indicador, respectivamente [17].

Essa característica presente no ambiente LabVIEW, de funcionalidades descritas através de ícones e não na forma textual usual, introduz uma importante diferença na forma como o fluxo de execução acontece em um dado trecho de código. Em códigos baseados em linguagens textuais instruções são executadas seqüencialmente na exata ordem em que aparecem. Em um diagrama, no entanto, onde as funcionalidades foram especificadas de forma gráfica, cada bloco ou elemento é executado no momento em que todas as entradas necessárias à sua execução encontram-se definidas. Essa característica bastante diferenciada pode ser muito útil na criação de trechos de código que devem ser executados em paralelismo aparente, em *threads* separadas, mas também pode apresentar alguma dificuldade quando se deseja atribuir certo sincronismo à alguns trechos do diagrama [17].

A figura 4 exemplifica a estrutura de um programa bastante simples destinado a realizar a soma de dois números fornecidos pelo usuário. No painel frontal do VI existem três elementos, dois deles utilizados para que o usuário forneça os valores a serem somados e um terceiro que abrigará o resultado uma vez que a execução tenha se encerrado. A figura 4 também traz o diagrama correspondente, onde é possível notar os elementos equivalentes aos presentes no painel frontal e a relação construída entre eles.

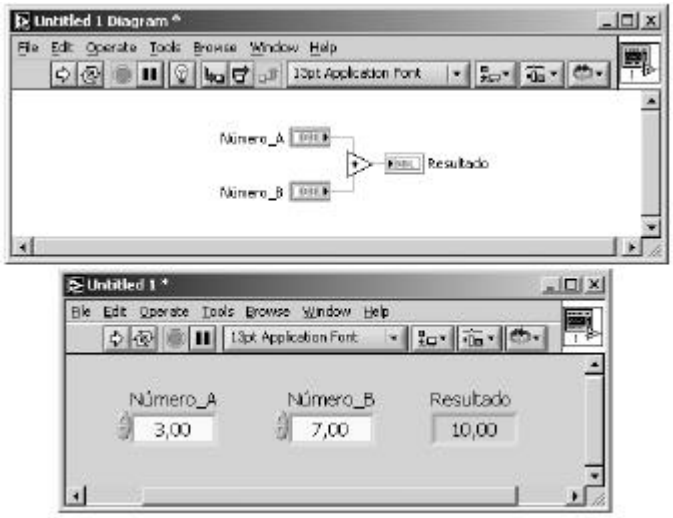


Figura 4 - Exemplo de painel frontal e diagrama de um simples VI.

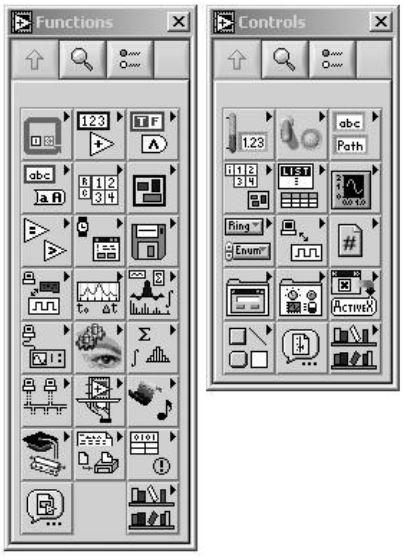


Figura 5 - Funções e controles de uma distribuição padrão.

Em adição às funcionalidades normalmente presentes, uma distribuição padrão do ambiente LabVIEW pode ser enriquecida com a instalação de ferramentas adicionais, fornecidas através de conjuntos denominados *toolkits*. Esse conjunto adicional de recursos inclui normalmente ferramentas especializadas em algumas tarefas como, por exemplo, projeto de filtros digitais, geração de relatórios, ferramentas avançadas de processamento digital de sinais, acesso a banco de dados e ferramentas de acesso aos serviços de Internet. O presente trabalho utiliza extensivamente os *toolkits* para acesso a banco de dados e as ferramentas de acesso aos serviços de Internet.

A figura 5 ilustra alguns dos elementos disponíveis ao programador em uma distribuição padrão do ambiente LabVIEW. A paleta da esquerda contém os elementos que podem ser inseridos no diagrama de um VI e englobam estruturas de controle, cantantes, e funções para as mais diversas finalidades. A paleta mostrada à direita da figura 5 abriga os elementos que podem ser alocados no painel frontal de um VI e contém, de forma geral, indicadores e controles.

2.3.1. CGI e LabVIEW

Nos parágrafos anteriores alguns aspectos relacionados à *Common Gateway Interface* foram apresentados, incluindo algumas de suas capacidades e a forma geral de seu funcionamento. A especificação da tecnologia CGI é genérica, no sentido em que os detalhes de implementação dependem fortemente do servidor HTTP utilizado. Dessa forma, os detalhes de implementação através do servidor HTTP disponível em ambiente LabVIEW, através das ferramentas do *Internet Toolkit*, são tratados nesta seção.

No instante em que um usuário solicita algum documento estático através de seu navegador, ele apenas fornece um endereço[1] apropriado e o servidor retorna o conteúdo desse documento. Entretanto, muitas vezes é necessário ao cliente fornecer outros dados ao servidor que não apenas o endereço de um documento estático e, nesses casos, a forma mais corriqueira de alcançar esse objetivo é através do uso de formulários. Um formulário nada mais é do que um documento HTML contendo alguns comandos específicos, que permitem a inserção de elementos tais como caixas de texto, caixas de seleção, botões, etc... O preenchimento desses campos não é, entretanto, a única forma de um usuário fornecer dados ao servidor. Isto também pode ser realizado através de *links* adequados ou através de imagens mapeadas, onde juntamente com a informação à respeito do clique recebido pela imagem são também enviadas as coordenadas onde o clique se originou.

Uma das formas de passagem de parâmetros do navegador para uma aplicação CGI, activex do servidor HTTP, é utilizando a chamada *URL-Encoded Parameter String*, que consiste em um lista de elementos separados pelo símbolo '&' cada um, por sua vez, contendo o nome de uma determinada variável e seu valor, separados pelo símbolo de '='. Parâmetros como o nome do usuário, sua idade e seu sexo seriam codificados em uma URL conforme ilustra o quadro seguinte.

```
Nome=Luciana%20Ribeiro&idade=25&sexo=F
```

[1] URL ou *Universal Resource Locator* é a denominação rigorosa para o caminho de um documento na Internet.

Uma aplicação CGI chamada `processa_dados.vi`, localizada no diretório ./`cgi-bin/` de um servidor HTTP executando na porta 80 da máquina 143.106.8.162 poderia receber os parâmetros acima através do seguinte endereço:

```
http://143.106.8.162:80/cgi-
bin/processa_dados.vi?nome=Luciana
%20Ribeiro&idade=25&sexo=F
```

Manter *links* estáticos com esse nível de complexidade e detalhamento é algo inviável. No entanto, através do emprego de formulários é possível gerar endereços como o apresentado de maneira que as informações fornecidas pelo usuário nos campos do formulário sejam transportadas até o servidor. Dessa forma, a aplicação CGI recebe como parâmetros de entrada as informações contidas no formulário juntamente com as chamadas variáveis de ambiente, produzidas pelo servidor HTTP e endereçadas mais adiante.

```
        <FORM>
<form name="fomulário_de_exemplo"
      action="/cgi-bin/processa_dados.vi"
      method="GET">
        ...
</form>
```

O quadro acima ilustra parte da construção de um formulário. Deste exemplo é possível observar três propriedades associadas ao formulário:

- **NAME**: Atribui um nome ao formulário sendo criado. Irrelevante para o tema discutido nessa seção, porém importante para permitir que elementos de código HTML externos ao formulário, *scripts* em JAVASCRIPT por exemplo, possam se referenciar a elementos do formulário, principalmente em situações onde possam existir mais de um formulário na mesma página.

- **ACTION**: Informa ao servidor HTTP o endereço, absoluto ou relativo, da aplicação CGI que deve ser ativada no momento em que o formulário for submetido.
- **METHOD**: Determina qual método GET ou POST será utilizado, ou mais precisamente, de que forma os dados serão enviados para o servidor. Utilizando o método GET os dados contidos no formulário são automaticamente codificados em uma URL, na forma exposta no início dessa seção e enviados para o servidor. O método POST, por sua vez envia as informações encapsuladas no cabeçalho do pedido HTTP. Do ponto de vista da aplicação CGI o método utilizado é indiferente já que existem elementos igualmente versáteis para o tratamento de solicitações originadas a partir de um ou outro método. Uma importante diferença, no entanto, e que deve orientar a escolha de um ou outro método é a natureza das informações sendo enviadas a partir do navegador. Informações que exijam certo sigilo em seu tratamento, como por exemplo, senhas, números de conta e informações pessoais do usuário, devem ser enviadas utilizando-se preferencialmente o método POST. Caso utilizado, o método GET tornaria as informações transmitidas parte da URL enviada ao servidor e, portanto, informações sensíveis poderiam ser armazenadas no cache ou no histórico do navegador, estando disponíveis a outros usuários. Isso comprometeria a segurança.

Além dos parâmetros gerados a partir das entradas do usuário, o aplicativo CGI também recebe uma série de outros parâmetros, automaticamente produzidos pelo servidor HTTP, denominados variáveis de ambiente. Através dessas variáveis é possível a uma aplicação CGI ter acesso sobre algumas características da sessão HTTP corrente, tais como endereço IP do cliente, o tipo de navegador utilizado, versões de software, etc...

A tabela 3 abaixo contém algumas variáveis de ambiente geradas pelo servidor HTTP presente no *toolkit* utilizado, juntamente com os valores que cada uma costuma assumir.

GATEWAY_INTERFACE	CGI/1.1
SERVER_SOFTWARE	G_Web_Server/5.0
SERVER_NAME	127.0.0.1
SERVER_PORT	80
DOCUMENT_ROOT	/C/Program Files/National Instruments/LabVIEW 6.1/internet/home
REMOTE_HOST	127.0.0.1
REMOTE_ADDR	127.0.0.1
REMOTE_PORT	1176
REQUEST_METHOD	GET
SERVER_PROTOCOL	HTTP/1.1
HTTP_REFERER	http://localhost/examples/environ.htm
HTTP_USER_AGENT	Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0)
HTTP_ACCEPT	image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/vnd.ms-powerpoint, application/vnd.ms-excel, application/msword, */*
HTTP_CONNECTION	Keep-Alive
REMOTE_USER	Utilizada apenas com esquemas de autenticação
REMOTE_IDENT	Utilizada apenas com esquemas de autenticação
QUERY_STRING	Contém os parâmetros enviados através da URL

Tabela 3 - Variáveis de ambiente passadas à aplicação CGI.

Por fim, a última possibilidade para que um usuário forneça entradas à uma aplicação CGI através de seu navegador é a técnica que se baseia em imagens mapeadas. Essa técnica pode ser aplicada de duas formas diferentes. A primeira delas associa a uma dada imagem um mapa que a subdivide em diversas regiões, estando cada uma destas associada a uma URL diferente. O quadro abaixo ilustra o procedimento, onde regiões circulares e retangulares são produzidas.

```
<IMG SRC="../imagens/imagem.gif" BORDER="0" USEMAP="#formas">

<MAP Name="formas">
<AREA Shape="circulo" coords = "150,50,30"
   HREF="circulo.htm">
<AREA Shape="retangulo" coords = "30,20,91,71"
   HREF="retangulo.htm">
</MAP>
```

Alternativamente, o mapeamento pode permanecer em um arquivo no lado do servidor. Esse arquivo deve possuir conteúdo semelhante ao exemplificado no quadro abaixo.

```
rect    rect.htm    31,20 92,71
circle  circle.htm  150,50 180,50
poly     poly.htm     20,110  40,90  90,90  110,110
90,130  39,130  20,110  20,110  20,110
```

Nesse caso o código inserido no arquivo transferido para o cliente deve se assemelhar ao mostrado no quadro seguinte.

```
<A HREF="formatos.map">
<IMG SRC="../imagens/shapes.gif" BORDER="0" ISMAP
</A>
```

A segunda técnica para o mapeamento de imagens utiliza o código exemplificado abaixo.

```
<A HREF="/cgi-bin/exemplos/image_map_cgi.vi">
<IMG SRC="../imagens/imagem.gif" BORDER="0" ISMAP>
</A>
```

Essa abordagem, que lança mão do atributo ISMAP, realiza uma chamada à aplicação CGI que receberá, como parâmetros codificados na URL, as coordenadas onde ocorreu o clique. Dessa forma cabe a essa aplicação a interpretação da coordenada recebida e a geração de código HTML adequado como resposta ao usuário.

Dentre as técnicas de mapeamento de imagem apresentadas, a última delas se destaca como a mais indicada para a implementação de interfaces com o usuário voltadas à experimentação remota, visto que é possível disponibilizar imagens reais dos painéis frontais do equipamentos utilizados e realizar, através de uma aplicação em CGI, a tradução das coordenadas recebidas em comandos reais enviados para os equipamentos. A riqueza de regiões ativas que essas imagens normalmente possuem torna mais difícil a utilização das técnicas convencionais de mapeamento de imagens inicialmente apresentadas, porém não as inviabiliza como possíveis soluções.

Uma vez endereçadas as técnicas que permitem que as informações provenientes do cliente cheguem até uma dada aplicação CGI, os parágrafos seguintes discutem a estrutura interna dessas aplicações em LabVIEW.

Toda aplicação CGI desenvolvida em LabVIEW deve apresentar invariavelmente a estrutura apresentada abaixo.

Código 1 - Modelo de código CGI em LabVIEW.

O diagrama acima permite identificar três regiões de interesse de uma aplicação CGI. A primeira delas, situada à esquerda é responsável por receber todos os parâmetros discutidos até o momento, entradas do usuário e variáveis de ambiente e disponibilizá-las para a região central do diagrama. Essa segunda região de interesse deve conter todo o código da aplicação, que deve depender em algum grau dos parâmetros processados pela primeira parte. Por fim, a região a direita no código 1 é utilizada para que os resultados produzidos pela aplicação possam ser retornados ao navegador e exibidos ao cliente.

Os elementos constituintes do código 1, por serem parte intrínseca das aplicações CGI, são discutidos em maior detalhe na tabela abaixo.

A tecnologia CGI e a forma pela qual é implementada em ambiente LabVIEW apresentam inúmeros outros aspectos e possibilidades que não foram tratados nessa seção por não estarem diretamente relacionados, ou não terem sido diretamente aplicados no presente trabalho. Entre esses tópicos está, por exemplo, a capacidade de implementar persistência através da utilização dos chamados *cookies*.

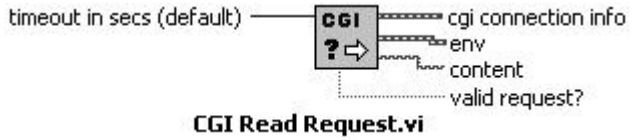
	Aguarda o valor indicado em **timeout**, ou até 60 segundos pela chegada de uma requisição de execução válida. (60 segundos é o valor padrão de tempo pelo qual o servidor mantém uma aplicação CGI em *cache*.) Se, durante esse período, uma solicitação de execução válida for disparada, ficam disponíveis nos terminais de saída os parâmetros enviados pelo usuário e as variáveis de ambiente produzidas pelo servidor. O terminal **cgi connection info** é uma estrutura que identifica uma sessão CGI em particular.
	Escreve a saída do processamento realizado pela aplicação CGI de volta ao servidor, que as repassará ao navegador do cliente. O terminal de entrada **header** é utilizado nas situações onde o conteúdo do terminal de entrada **content** não é código HTML simplesmente.
	Informa ao servidor HTTP que a aplicação CGI encerrou sua execução e pode ser descarregada da memória do sistema.

Tabela 4 - Principais elementos de uma aplicação CGI.

2.3.2. Esquemas de Autenticação de Usuário

A discussão sobre a interface remota adotada, tratada na seção 2.2, levanta como aspecto negativo do ponto de vista de segurança, a possibilidade de que qualquer usuário, munido de um navegador e de uma conexão com a Internet, possa obter acesso aos recursos disponibilizados pelo sistema. Adicionalmente, a base de dados implantada e discutida na seção 2.4 seguinte, prevê elementos que permitam avaliar a forma de utilização dos recursos do laboratório por cada usuário.

Os argumentos apresentados no parágrafo anterior, juntamente com a natureza dos equipamentos disponibilizados, tornam fundamental a introdução de algum mecanismo de autenticação que torne possível apenas a usuários previamente cadastrados e identificados o acesso ao recursos remotos.

A apresentação e discussão de algumas das possibilidades existentes para a implementação de tais mecanismos é alvo desta seção, onde também são enfocadas os aspectos de implementação do método escolhido em ambiente LabVIEW.

O processo de autenticação de um usuário através de seu nome de usuário e de sua senha pode ser dividido em três etapas. A primeira delas consiste no recolhimento das informações do usuário pelo navegador, sua transmissão até o servidor e sua validação junto à base de dados de usuários autorizados. A segunda etapa, pela ausência de uma conexão persistente entre cliente e servidor, consiste na implementação de mecanismos capazes de manter o estado de usuário autorizado, para que este não seja forçado a revalidar sua identidade a cada nova página visitada. Por fim, a última etapa envolvida deve fornecer meios ao usuário de encerrar sua autenticação junto ao servidor, informando a este que as informações a respeito do estado daquele podem ser destruídas.

Os documentos de padronização do protocolo HTTP, [18][19][20] prevêem dois métodos de autenticação de usuário, que devem serem suportados por todos os navegadores, denominados "*digest authentication*" e "*basic authentication*".

O protocolo HTTP passou a oferecer suporte ao método *digest authentication* com o objetivo de evitar que os nomes e senhas dos usuários fossem transmitidos em forma de texto puro entre cliente e servidor, pois isso permite que essas informações sejam capturadas por terceiros executando algum tipo de *sniffer*[2]. Embora se trate de um método devidamente especificado e padronizado, nem todos os servidores HTTP e navegadores o suportam. Além disso, alguns navegadores e servidores o implementam de forma específica, introduzindo sérias restrições à sua aplicabilidade. Embora esses argumentos já sejam fortes o bastante para que esse método não seja o empregado, os parágrafos seguintes discutem brevemente seu funcionamento, pois o método pode se tornar alternativa interessante em futuros trabalhos.

O mecanismo de autenticação tem seu início através da solicitação de algum documento do navegador ao servidor. O servidor, por sua vez, ao identificar que se trata de um

[2] *Sniffers* são aplicativos normalmente utilizados por gerentes de rede para a análise do tráfego em um determinada rede de computadores. Interagindo diretamente com o adaptador de rede presente, esses aplicativos capturam todos pacotes que circulam pela rede, podendo registrar senhas e nomes de usuário que estejam sendo enviados sem mecanismos de criptografia.

documento onde incidem restrições de acesso, nega a solicitação enviando de volta ao navegador a informação de que o método *digest* de autenticação é necessário para a obtenção do documento, juntamente com uma seqüência de caracteres conhecida como "*nonce*". Este, por sua vez, depende da hora da solicitação e do endereço do cliente entre outras variáveis, o que a torna diferente a cada nova requisição. O navegador solicita então ao usuário seu nome de usuário e sua senha. Essas duas seqüências de caracteres são concatenadas e enviadas como entrada para o algoritmo MD5[3] que produz um *checksum* dessa seqüência de caracteres. Paralelamente, o endereço do documento solicitado é concatenado ao método utilizado em sua solicitação, GET ou POST e, também através do algoritmo MD5, gera outro *checksum*. Por fim, esses dois *checksums* calculados são concatenados com o *nonce* inicialmente enviado pelo servidor e servem de entrada novamente para o algoritmo MD5. O *checksum* resultante dessa última etapa é efetivamente retornado ao servidor, com a nova solicitação pelo documento original, juntamente com o nome do usuário em sua forma original e o valor de *nonce*.

Esta nova solicitação chega até o servidor que busca em seu banco de dados pelo nome de usuário, obtendo assim sua senha verdadeira. O servidor calcula os mesmos *checksums* computados pelo navegador e compara com os valores recebidos na nova solicitação. Dessa forma, é possível a autenticação do usuário sem a transmissão de informações sensíveis diretamente pela rede.

O segundo método de autenticação definido pelo protocolo HTTP é o método chamado "*basic authentication*". O funcionamento deste esquema de autenticação é em parte similar ao descrito para o método anterior. O navegador solicita um determinado documento ao servidor que, por se tratar de documento de acesso restrito, o rejeita e informa ao navegador a necessidade de uma combinação de nome de usuário e senha. O navegador obtém junto ao usuário essas informações e responde ao servidor enviando de volta a solicitação pelo documento. Nessa segunda tentativa o servidor verifica as informações recebidas e prossegue

[3] O algoritmo MD5 recebe seqüências de caracteres de comprimento arbitrário e gera como saída um *checksum* 16 bytes. O algoritmo MD5 foi concebido de forma que dado um *checksum*, encontrar o bloco de texto que o originou é uma tarefa extremamente custosa computacionalmente e, dessa forma, é considerado um algoritmo de criptografia de mão única.

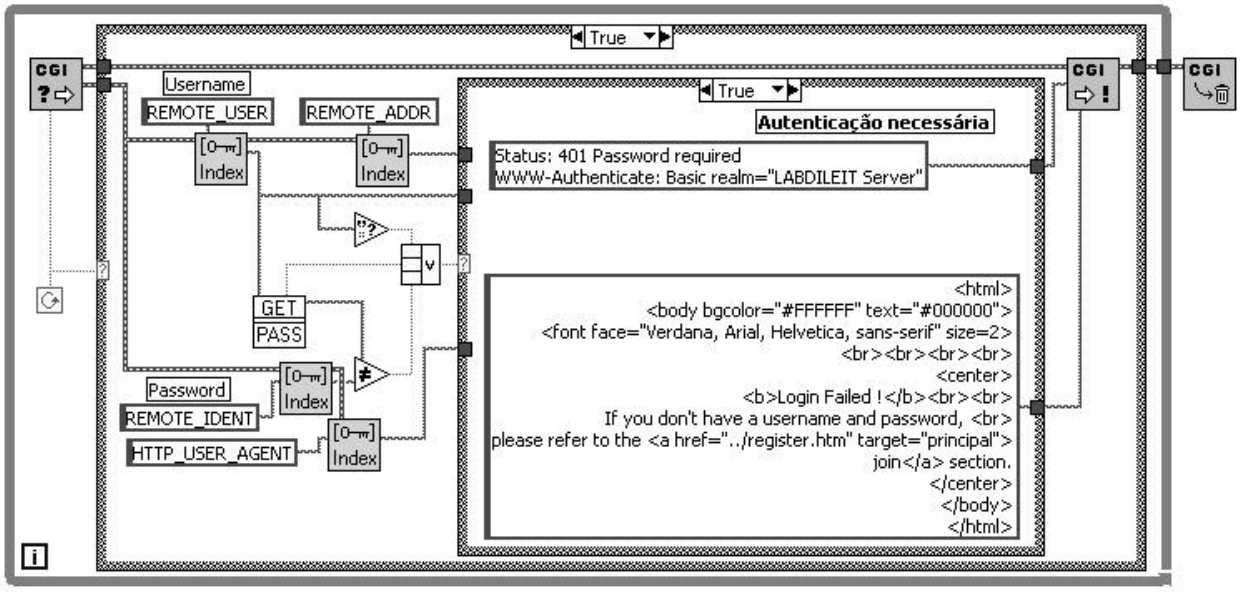
com o envio do documento, caso nome de usuário e senha estejam corretos. O servidor também configura variáveis de ambiente que abrigam os valores correntes de nome de usuário e senha, que se tornam disponíveis para as aplicações CGI.

Teoricamente, todo o processo descrito anteriormente deve ser repetido para o acesso de cada página de acesso restrito. Isso levaria à necessidade de duas requisições por página, pois a primeira seria rejeitada e somente a segunda traria as informações acerca do nome do usuário e sua senha. Na prática, entretanto, a maioria dos navegadores armazena essas informações e as acrescenta de forma automática nas requisições seguintes, de forma imperceptível ao usuário.

O método de autenticação descrito anteriormente e eleito para o laboratório remoto abordado neste trabalho possui alguns aspectos negativos que merecem ser discutidos em mais detalhe. O primeiro deles é o fato de que todas as informações utilizadas no processo de autenticação são enviadas pela rede de forma pura, tornando possível que essas informações sejam capturadas, conforme citado durante a apresentação do método *digest* de autenticação. Outro ponto é a impossibilidade do usuário informar ao navegador que sua sessão como usuário chegou ao fim e que, a partir daquele instante seu nome de usuário e sua senha não mais devem ser enviados para o servidor junto com novas requisições por documentos. A única forma de se alcançar esse estado é reinicializando o navegador. NO entanto, isto faz com que sejam comuns os avisos, onde esse método de autenticação é empregado, solicitando ao usuário que se certifique de ter fechado todas as janelas ativas de seu navegador após a consulta aos documentos protegidos. Por fim, outro aspecto negativo do método *basic* de autenticação relaciona-se com a falta de controle sobre a aparência da caixa de diálogo exibida ao usuário no instante de recolhimento de seu nome de usuário e senha. Essa janela está estreitamente ligada ao navegador utilizado.

A implementação do procedimento de autenticação de usuário segundo o método *basic*, em ambiente LabVIEW é realizado de acordo com o código 2 ilustrado abaixo. O diagrama representado segue a estrutura clássica de uma aplicação CGI, onde as variáveis de ambiente *REMOTE_USER* e *REMOTE_IDENT* são verificadas. Num momento inicial essas variáveis de ambiente encontram-se vazias e, neste caso, a aplicação retorna no cabeçalho do documento de resposta código que informa ao navegador a necessidade de um nome de usuário e de uma senha. O navegador então fornece ao usuário sua janela particular para o recolhimento

dessas informações, tal como mostra a figura 6, e exibirá, em caso de cancelamento dessa janela por parte do usuário, o documento representado pelo código enviado no corpo do documento de resposta, ilustrado através da figura 7.



Código 2 - Procedimento de autenticação de usuário.

Figura 6 - Janela de autenticação produzida pelo navegador.

Figura 7 - Resultado de uma autenticação inválida.

2.4. Banco de Dados do Sistema

A disponibilização remota de equipamentos e componentes exige alguma forma de controle sobre sua utilização, de maneira a preservá-los de acidentes e de usuários maliciosos. Formas de proteção contra circuitos mal dimensionados são discutidas na seção 2.6, dedicada ao controle dos equipamentos. Assim, nesta seção somente será discutida a implementação de um sistema de banco de dados para armazenar informações sobre os usuários e sobre seus acessos ao laboratório. A introdução de uma base de dados adequada permite a capacidade de analisarmos estatisticamente não apenas o perfil dos usuários recebidos pelo laboratório como também a forma como os recursos vêm sendo utilizados com o passar do tempo.

2.4.1. Estrutura Geral

Um sistema de banco de dados foi adotado, prevendo-se a necessidade de cadastramento prévio para a utilização dos recursos do laboratório, juntamente com o interesse em poder analisar a forma como esses recursos vêm sendo utilizados.

A especificação da estrutura desse banco de dados considera não apenas a natureza das informações envolvidas, mas também a forma como se relacionam. Dessa forma, para que os dados de cadastramento de cada usuário bem como as informações sobre cada um de seus acessos sejam armazenadas, tornam-se necessárias duas tabelas. Uma denominada INFO_ACESSO e outra INFO_USUÁRIO. Conforme ilustra a figura 8, suas estruturas registram as informações sobre os acessos e dados de contato do usuário, de acordo com as suas respectivas denominações.

A estrutura adotada para as tabelas utiliza o campo *login*, comum às duas tabelas, como chave primária para o estabelecimento da relação 1:N existente entre os registros das tabelas INFO_USUARIO e INFO_ACESSO, respectivamente. O campo *login*, ao qual a chave primária do banco de dados é associada, identifica de forma unívoca o usuário na tabela INFO_USUÁRIO sendo utilizada posteriormente para a identificação de todos os acessos realizados por esse usuário na tabela INFO_ACESSO.

INFO_ACESSO				
login	data	arquivo	endereço	navegador

INFO_USUÁRIO							
Login	password	nome	sobrenome	data	e-mail_1	e-mail_2	...

INFO_USUÁRIO (continuação)						
...	e-mail_3	endereço	cidade	estado	CEP	país

Figura 8 - Estrutura de campos das tabelas adotadas no banco de dados.

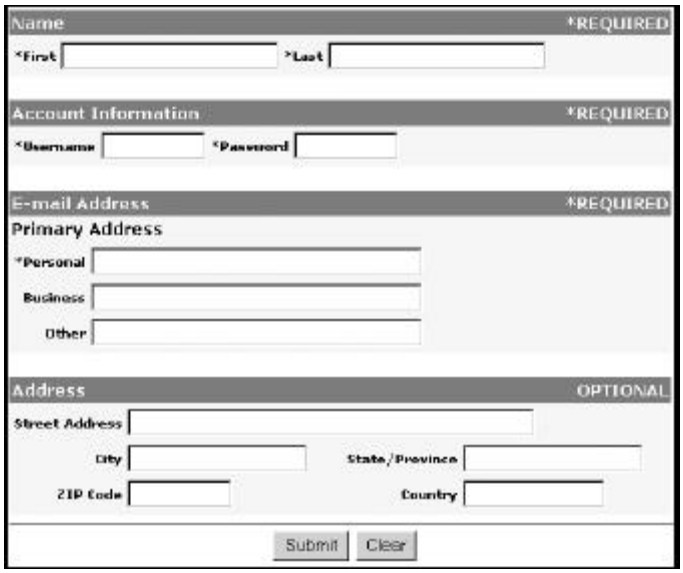


Figura 9 - Página para o cadastro de novos usuários.

2.4.2. A coleta de dados, seu tratamento e armazenamento

Os dados de cadastramento de um novo usuário são obtidos através do formulário HTML ilustrado na figura 9. Nesse formulário o usuário deve fornecer dados, tais como nome, sobrenome, endereços eletrônicos, etc..., que serão armazenados na tabela INFO_USUÁRIO.

Os campos assinalados — através de (*) — na figura anterior como sendo de preenchimento obrigatório são verificados pelo seguinte trecho de código em JAVASCRIPT.

```
<SCRIPT LANGUAGE="JavaScript">
function Valida() {
if ((document.join_form.first_name.value=="")||
    (document.join_form.last_name.value=="")||
    (document.join_form.username.value=="")||
    (document.join_form.user_password.value=="")||
    (document.join_form.personal_email.value=="")||
    (document.nda_form.checked==false)){
       alert("One or more required fileds were not corect filled.\n
              Required fields are identified with a *.\n\n
              Please try again.")
       return (false)
       }
else{
       return (true)
       }
}
</script>
```

Código 3 - Script de validação para cadastramento de novo usuário.

Uma vez realizada a verificação do preenchimento dos campos obrigatórios o conteúdo do formulário é enviado através do método POST para o servidor, conforme sugere o *link* no quadro abaixo, onde um VI específico para a recepção e tratamento dessas informações é colocado em execução.

```
<form   method="POST"   action="/cgi-bin/join.vi"   name="join_form"
onSubmit="return Valida()">
```

Esse VI, denominado *join.vi*, recebe essas informações e as armazena na base de dados, gerando como resposta código HTML dependente do resultado dessa operação de inserção. Caso o *login* escolhido já esteja sendo utilizado por outro usuário a página produzida solicita ao usuário uma nova tentativa, dessa vez com outro *login*. Caso contrário, é exibida uma página confirmando o cadastramento do usuário. Em ambos os casos o usuário é remetido à página principal do laboratório após alguns segundos.

Login válido.	Login inválido.
<HTML> <HEAD> <META CONTENT="15; URL=../home.htm" HTTP-EQUIV="Refresh"> <meta HTTP-EQUIV="Content-Type" CONTENT="text/html; CHARSET=iso-8859-1"> </HEAD> <BODY BGCOLOR="#FFFFFF" TEXT="#000000"> <FONT FACE="Verdana, Arial, Helvetica, sans-serif" size="-1"> The login you chose is already being used. Please return to the previous form and try again with a different one. </FONT> </BODY> </HTML>	<HTML> <HEAD> <META CONTENT="15; URL=../home.htm" HTTP-EQUIV="Refresh"> <meta HTTP-EQUIV="Content-Type" CONTENT="text/html; CHARSET=iso-8859-1"> </HEAD> <BODY BGCOLOR="#FFFFFF" TEXT="#000000"> <FONT FACE="Verdana, Arial, Helvetica, sans-serif" size="-1"> You were successfully added to the LABDILEIT database. You will receive an e-mail from the LABDILEIT adminisrtator confiming your subscription. Only then you will be able to use the laboratory. Thank you for registering ! <A HREF="../home.htm" TARGET="principal"> Click here</A> to get back. </FONT> </BODY> </HTML>

Código 4 - Código HTML resultante de um novo cadastramento.

O código responsável pelo recebimento e armazenamento dos dados de cadastro de um novo usuário é apresentado através do código 5 seguinte. A análise desse código permite a identificação, além da estrutura clássica para a recepção de dados enviados por uma página ou formulário, sub-VIs específicos para o acesso à banco de dados.

O acesso a bancos de dados é realizado em ambiente LabVIEW através do *Database Connectivity Toolkit* que adiciona a série de VIs mostrados na figura 10 ao conjunto padrão de funções presentes no LabVIEW 6.1.

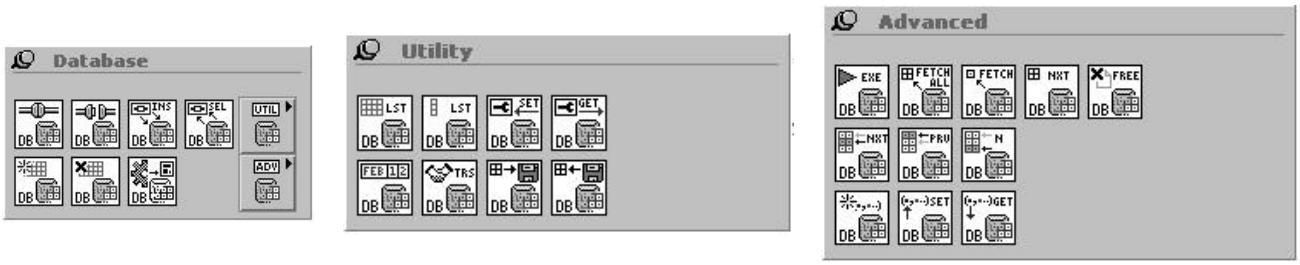
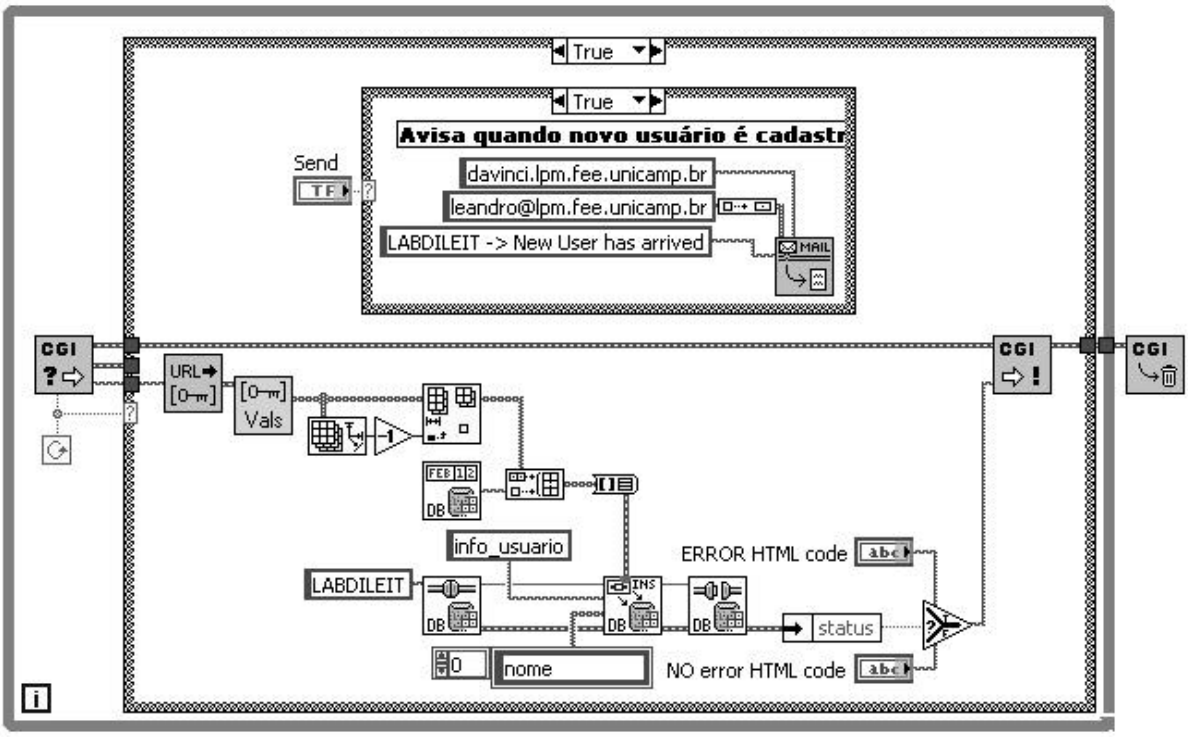


Figura 10 - Ferramentas de acesso a banco de dados.



Código 5 - Diagrama do VI *join.vi*.

Dentre as várias novas funcionalidades adicionadas pela instalação do *Database Connectivity Toolkit*, aquelas que são fundamentais para a operação de inserção de um novo registro num banco de dados são detalhadas na tabela abaixo.

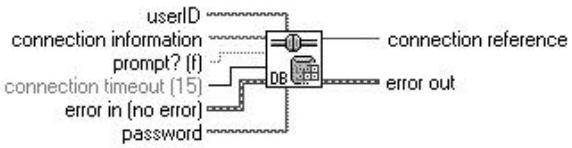
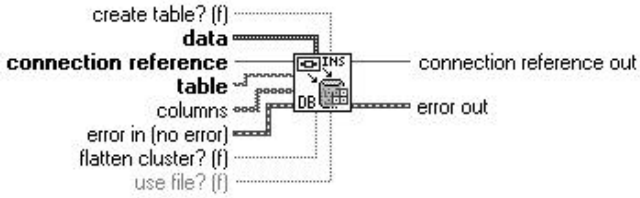
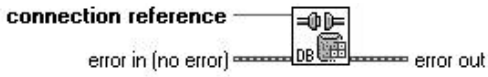
	Estabelece a conexão com o banco de dados, recebendo como principal argumento o nome DSN — *Data Source Name* — registrado no gerenciador ODBC — *Open Database Connectivity*.
	Insere novos registros no banco de dados atualmente conectado. Devem ser especificadas a tabela e suas colunas onde as informações devem ser inseridas.
	Encerra a conexão com o banco de dados.

Tabela 5 - Algumas das funcionalidades do *Connectivity Toolset*.

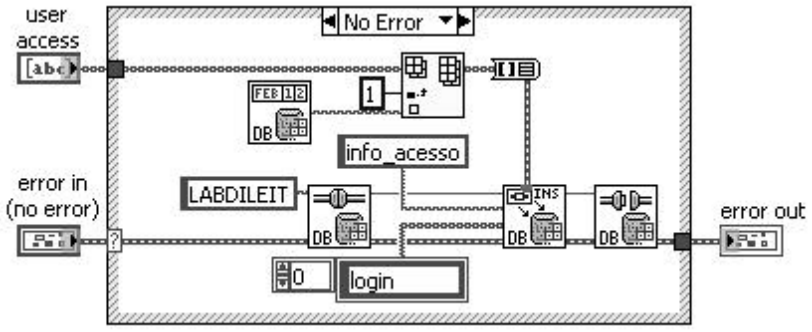
Para a correta utilização desses novos recursos, algumas configurações devem ser realizadas de forma a permitir o acesso eficiente e independente do gerenciador de banco de dados utilizado, conforme apresentado no apêndice A.

Opcionalmente, no diagrama anterior, foi incluída a possibilidade de envio de um e-mail de notificação ao administrador do laboratório quando um novo usuário realiza seu cadastro no sistema.

2.4.3. O registro de acessos.

O registro de acesso de cada usuário ao laboratório é, conforme citado anteriormente, registrado em uma tabela específica de estrutura descrita na figura 8 e denominada INFO_ACESSO.

No momento da recepção de um novo arquivo contendo a descrição do circuito enviado pelo usuário; seu *login*, a data e hora da recepção, o endereço IP desse usuário e o navegador utilizado para o envio são registrados na tabela INFO_ACESSO pelo VI denominado *add_access.vi*. Seu código é apresentado abaixo e possui estrutura muito semelhante ao descrito anteriormente para o VI *join.vi*. Deve-se ressaltar, entretanto, que desta vez o nome da tabela e sua estrutura devem ser adequados.



Código 6 - Código do VI *add_access.vi*.

2.4.4. A geração de relatórios.

Ao usuário do laboratório é oferecida a possibilidade de consultar seus dados de cadastro além de seu histórico de utilização do laboratório. Para que isso seja possível é necessária a busca nas duas tabelas que integram o banco de dados do sistema. A busca é realizada em ambas as tabelas através do campo *login*, que conforme citado em parágrafos anteriores é a chave primária da tabela INFO_USUARIO.

Essa busca é realizada através do envio de um comando na linguagem padrão de acesso a banco de dados SQL – **S**tructured **Q**uery **L**anguage – implementada através da função apresentada na tabela abaixo.

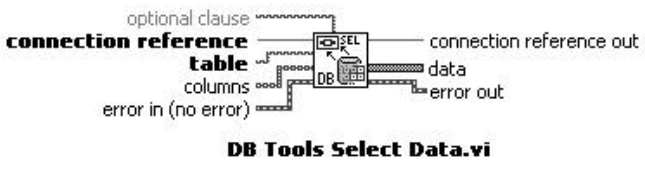
	Seleciona as colunas especificadas dos registros presentes na tabela fornecida como entrada. Filtros podem ser fornecidos segundo o padrão SQL para a realização de buscas específicas.

Tabela 6 - Função de busca em base de dados.

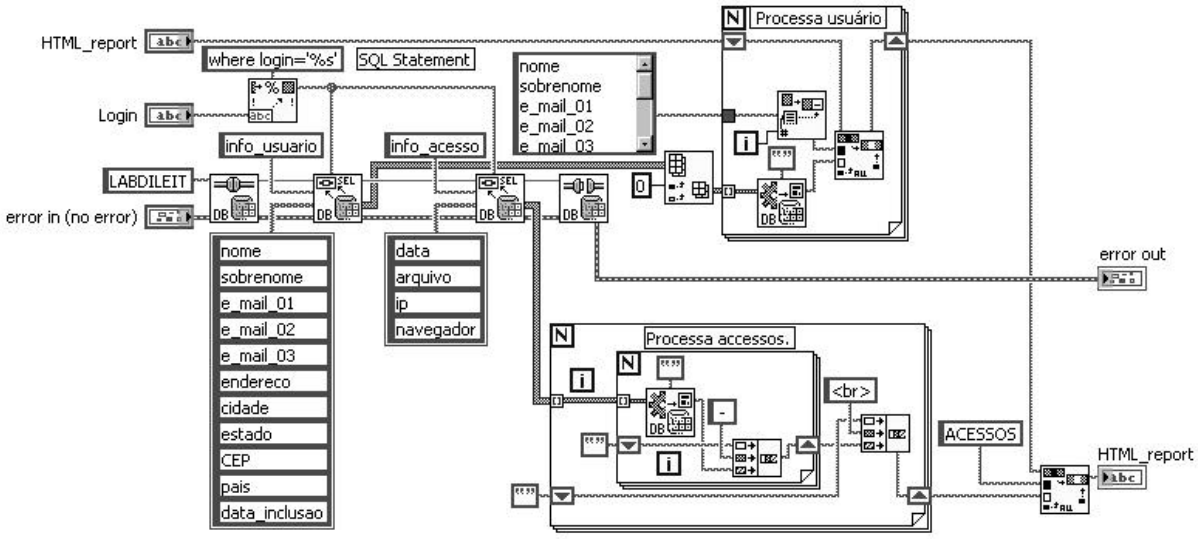
O código 7, apresentado a seguir, é responsável pela realização da consulta à base de dados e geração do código HTML correspondente. Da mesma forma que os outros acessos à base de dados para a inserção de elementos, o fluxo desse trecho de código também se inicia pela conexão com o DSN. Em seguida são enviados dois comandos SQL na forma:

```
SELECT colunas FROM tabela WHERE login = '<login>'
```

Como resposta a esse comando é retornado apenas um registro da tabela INFO_USUARIO, pois a busca foi realizada através de sua chave primária. A tabela INFO_ACESSO, por sua vez, pode retornar qualquer quantidade de registros como resposta. Ambas as respostas são então tratadas e integradas em um modelo de código HTML dando origem à página que retorna ao usuário.

O banco de dados do laboratório foi estruturado utilizando o gerenciador de banco de

dados Microsoft Access 2000™. Entretanto, devido ao emprego da interface ODBC, cuja definição e configuração é abordada no apêndice A, a migração para outro sistema de gerenciamento de banco de dados pode ser realizada sem que o código em LabVIEW tenha de experimentar qualquer alteração.



Código 7 - Código do VI *generate_report.vi*.

2.5. O envio do arquivo de NETLIST

Conforme descrito na seção 2.1 a interação entre o usuário e o laboratório remoto têm seu início com o envio do arquivo contendo a descrição topológica do circuito sob análise. Esta seção apresenta o mecanismo utilizado para o envio desse arquivo bem como os procedimentos que permitem sua correta recepção pelo servidor.

Os formulários HTML permitem que informações fornecidas pelo usuário sejam recolhidas através de uma página. Entretanto, sob alguns aspectos, isso ocorre de forma limitada, principalmente quando há necessidade do envio de arquivos de dados [21]. Essa fragilidade levou à inclusão de um elemento novo à especificação da linguagem HTML, que definia, até então, apenas oito elementos para a entrada de dados em um formulário[4]: CHECKBOX, HIDDEN, IMAGE, PASSWORD, RADIO, RESET, SUBMIT e TEXT.

[4] Considera-se aqui apenas as possibilidades para o atributo *TYPE* de um elemento do tipo *INPUT*.

Através desse novo elemento, denominado FILE é possível ao usuário a navegação e a seleção de um dado arquivo em seu computador. A forma de codificação utilizada para os formulários onde está presente esse novo elemento deve ser alterada para *"multipart/form-data"*.

A figura 11 abaixo exibe o formulário utilizado pelo usuário para o envio de seu *netlist.* Através do botão *browse* é possível selecionar o arquivo desejado e enviá-lo clicando em seguida em *submit.*

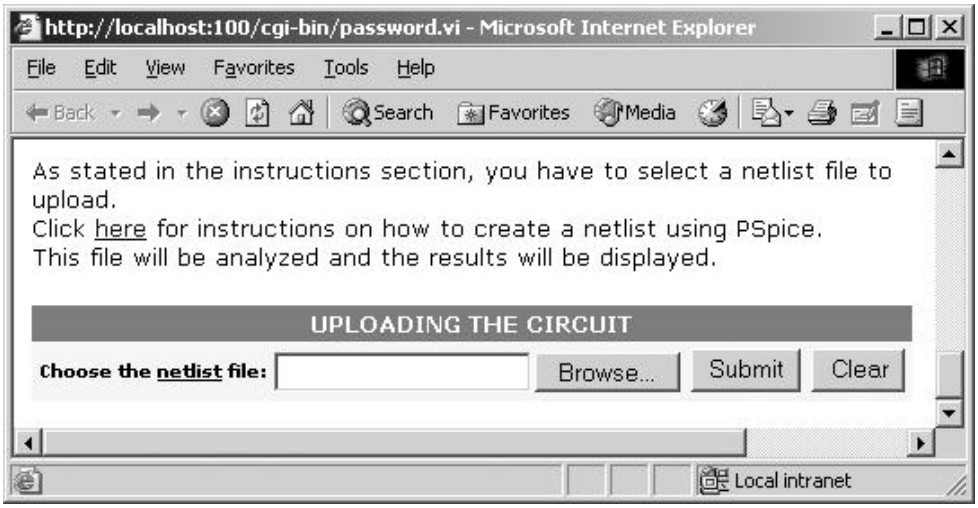


Figura 11 - Formulário para o envio do arquivo de netlist.

O quadro abaixo corresponde ao código HTML utilizado para a geração do formulário acima. No código aparecem em destaque o método de codificação utilizado e a sintaxe utilizada para a criação do elemento file.

```
<FORM  NAME="upload"
       ACTION="/cgi-bin/upload_file.vi"
       METHOD="post"
       ENCTYPE="multipart/form-data"
       onSubmit="return valida();">

<table border=0 cellpadding=2 cellspacing=0 width=80%>
<tr bgcolor="#0099CC">
<td nowrap align="center">
<font face="Verdana, Arial, Helvetica, sans-serif" size=2 color="#ffffff">
<b>UPLOADING THE CIRCUIT </b></td>
</tr><tr bgcolor=#EFF7FF><td colspan=2><table border=0 cellpadding=2 cellspacing=0><tr>
<td nowrap align="right"><font face="Verdana, Arial, Helvetica, sans-serif" size=1>
<b>Choose the <u>netlist</u> file:</b></font></td><td nowrap align="right">

<INPUT TYPE="file" NAME="netlist">
<INPUT TYPE="submit" NAME="Submit" VALUE="Submit">
<INPUT TYPE="reset" NAME="Reset" VALUE=" Clear"

</td></tr></table>

</FORM>
```

Código 8 - Trecho de código HTML para envio de arquivo.

No quadro seguinte é mostrado o *script* utilizado para validar o arquivo selecionado pelo usuário. Na referência [21] é previsto um atributo denominado *ACCEPT* para o elemento *FILE*, que é responsável por realizar exatamente a mesma função desempenhada pelo *script* abaixo, isto é, a de permitir que apenas arquivos de um determinado tipo possam ser enviados.

```
<SCRIPT LANGUAGE="JavaScript">
function valida() {
       if (document.upload.netlist.value.indexOf('.net')==-1)
       {
              alert("Please, remember that only\n *.net files are allowed.\n\n Please
correct and try again");
return false;
       }
       else return true;
}
</script>
```

Código 9 - Código para validação do nome de arquivo selecionado.

Do lado do servidor é necessário que a aplicação CGI, que recebe o conteúdo do formulário anterior, compreenda o método de codificação utilizado , isto é o método *multipart/form-data*.

A forma de codificação usualmente adotada para o envio de dados presentes em um formulário, conhecida como *application/x-www-form-urlencoded*, é ineficiente para a transmissão de quantidades maiores de dados. É o caso, por exemplo, quando se trata do envio de arquivos, principalmente quando os arquivos podem conter grande quantidade de caracteres não imprimíveis, como é o caso de arquivos binários.

O método de codificação *multipart/form-data* seleciona inicialmente uma pequena seqüência de caracteres, que não deve estar presente em nenhum dos campos ou arquivos sendo enviados pelo usuário através do formulário, que funcionará como separador. Cada campo do formulário é então enviado na ordem em que aparece no formulário, devidamente identificados através de seus atributos *NAME*.

O quadro abaixo ilustra o conteúdo recebido pela aplicação CGI no momento em que um arquivo de *netlist* é recebido. Através da variável de ambiente *HTTP_CONTENT_TYPE*, criada automaticamente pelo servidor, é possível identificar os caracteres que funcionam como delimitadores de cada um dos campos que formam o formulário. No caso particular ilustrado esse separador vale "`----------------------------7d33b338110296`".

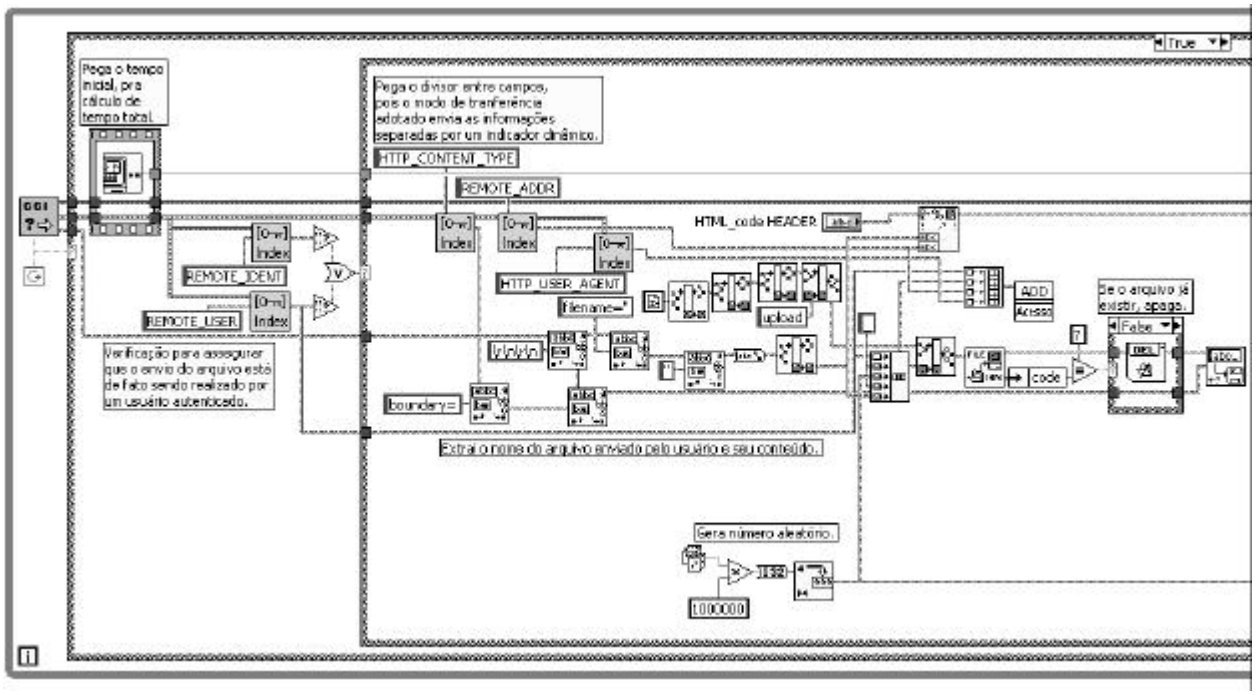
```
----------------------------7d33b338110296
Content-Disposition: form-data; name="netlist";
filename="C:\transistor.net"
Content-Type: application/octet-stream

GER1   0      1
CHANNEL3      0      1
R2     1      2
Q1     0      2      3
M2     3      4
R3     4      5
CHANNEL2      0      4
VDC2   0      5

----------------------------7d33b338110296
Content-Disposition: form-data; name="Submit"

Submit
----------------------------7d33b338110296--
```

O trecho de código ilustrado abaixo é responsável pela interpretação das informações mostradas no quadro anterior. Inicialmente o separador é identificado e, em seguida, o conteúdo do arquivo é extraído e salvo em um arquivo cujo nome é formado pelo login do usuário, um número de referência e o nome do arquivo enviado.



Código 10 - Trecho de código responsável pela recepção e gravação do netlist enviado.

2.6. O controle remoto dos equipamentos

A presente seção discute a forma como a interação do usuário com os equipamentos de medida e estímulo disponíveis é realizada, uma vez que o circuito enviado já se encontra corretamente estabelecido[5].

Em 1965 a empresa Hewlett-Packard desenvolveu um barramento dedicado à interconexão de sua linha de instrumentos programáveis, denominado *Hewlett-Packard Interface*

[5] A discussão detalhada da matriz de interconexão e de seu software de controle é assunto tratado no capítulo 3.

Bus — HP-IB. Sua relativa alta taxa de transferência, de até 1Mbyte/s, elevou sua popularidade e, em 1975, essa interface foi aceita como padrão pelo IEEE — *ANSI/IEEE Standard 488* [22] — que a renomeou para GPIB — *General Purpose Interface Bus*. Em 1987, um novo padrão — *ANSI/IEEE Standard 488.2* [23] — enriqueceu as definições originais para o barramento, definindo de forma mais precisa como a comunicação entre controladores GPIB e instrumentos deve ocorrer[6].

Toda a troca de informações entre o microcomputador servidor do sistema e os instrumentos disponíveis ocorre através de um barramento GPIB, considerando que todos os instrumentos utilizados permitem sua interconexão a esse barramento.

A interconexão de instrumentos deve obedecer algumas regras quanto à quantidade de instrumentos presentes e à máxima distância entre eles: não de deve utilizar mais de 15 instrumentos em um mesmo barramento, uma distância máxima de 4 metros entre dois instrumentos consecutivos, uma distância média de 2 metros entre instrumentos ao longo do barramento deve ser respeitada e, por fim, o comprimento total dos cabos que formam o

---

[6] Existem, entretanto, outras denominações utilizadas para esse barramento:

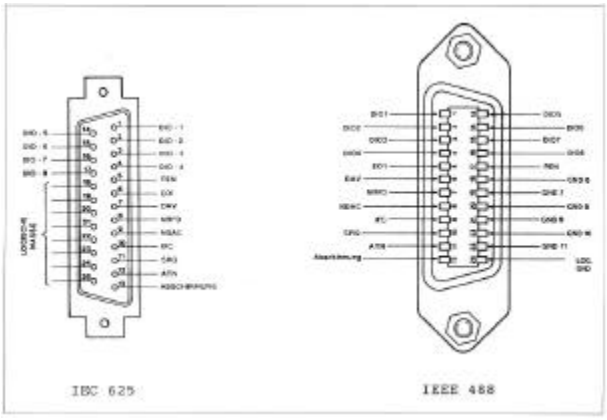
HP-IB – Hewlett-Packard Interface Bus.

GPIB – General Purpose Interface Bus.

IEC-625 – Denominação definida pelas normas internacionais e utilizada em países europeus.

IEEE-488 – Denominação definida pela norma americana.

Os conectores especificados pelas normas americana e européia diferem conforme ilustrado na figura ao lado. A diferença é, entretanto, apenas mecânica. O conector recomendado pela IEC625 possui 25 terminais, enquanto aquele recomendado pela IEEE488 possui um terminal a menos.



barramento não deve ultrapassar 20 metros.

Existem duas formas básicas possíveis para a interligação dos instrumentos em um barramento GPIB, conforme ilustrado na figura 12. Na primeira possibilidade, denominada ligação em estrela, as ligações para todos os instrumentos partem diretamente do controlador, enquanto na segunda possibilidade, denominada linear, a ligação a um instrumento parte do anterior. Também a construção de um barramento GPIB que contenha os dois tipos de interconexão é possível.

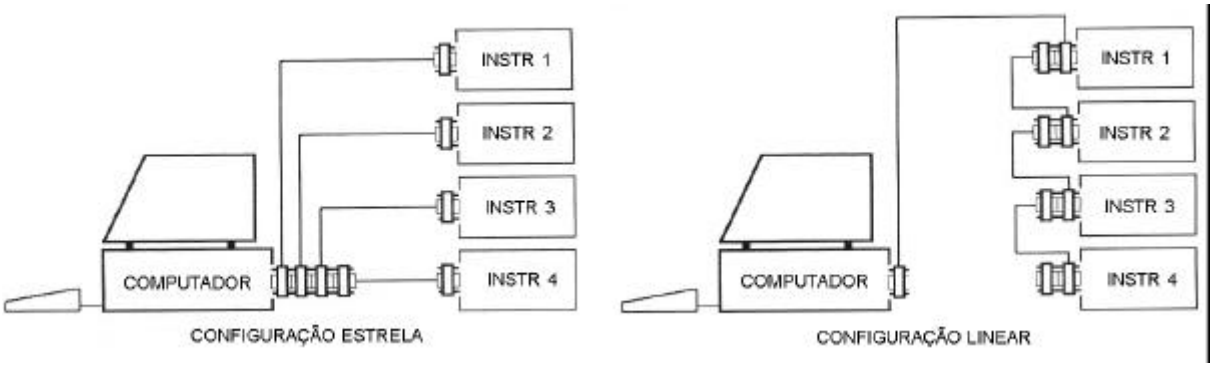


Figura 12 - Configurações possíveis para a interconexão de instrumentos.

A instrumentação controlada por computador evoluiu apreciavelmente nas últimas décadas, propiciando maior velocidade na transferência de dados, elevado nível de padronização das instruções utilizadas, além da introdução de novas camadas de software que tendem a diminuir o tempo necessário ao desenvolvimento de sistemas de instrumentação automatizados. As referências [24] e [25] são fontes indicadas para maiores informações sobre o barramento GPIB e as novas tecnologias a ele relacionadas, introduzidas nos últimos anos.

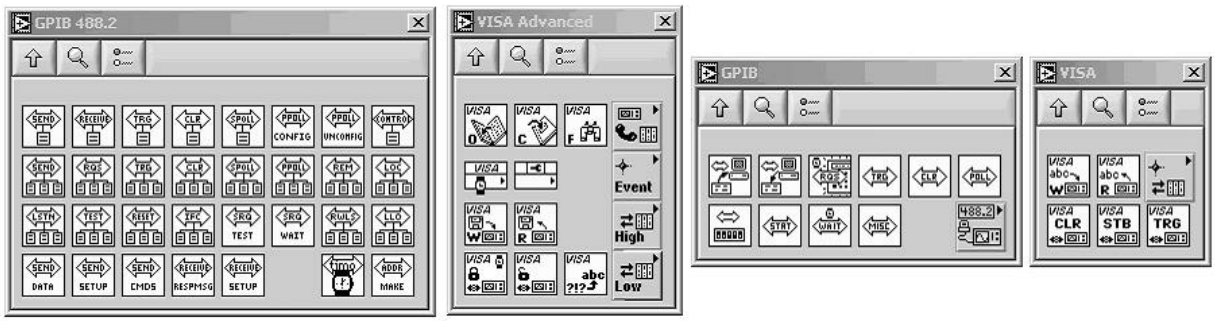


Figura 13 - Ferramentas disponíveis para comunicação GPIB.

O ambiente de desenvolvimento adotado provê uma série de ferramentas que permitem a interação com equipamentos de instrumentação, seja através de comandos GPIB tradicionais, ou utilizando tecnologias mais recentes, tais como VISA[7] ou IVI[8]. A figura 13 traz parte dessas ferramentas.

No laboratório remoto proposto nesse trabalho, o usuário tem a oportunidade de interagir com os instrumentos de medição e estímulo após o envio e a montagem de seu circuito. Essa tarefa é realizada através do formulário da figura 14, exibido ao usuário após a correta interconexão dos terminais dos componentes utilizados pela matriz de interconexão.

Através do formulário o usuário dispõe de meios para controlar os equipamentos disponíveis:

- 1 Voltímetro - Multímetro HP3478A.
- 1 Amperímetro - Multímetro HP3478A.
- 4 Canais de Osciloscópio HP54503.
- 1 Gerador de Sinais HP8904A.
- 2 Fontes de polarização TEK PS2520G.

Após a definição dos parâmetros de configuração para cada um dos equipamentos utilizados em seu experimento essas informações são submetidas ao servidor através do botão *UPDATE Measurements*. No lado do servidor essas configurações são recebidas e passadas à aplicação CGI responsável pelo interfaceamento com os equipamentos. Essa aplicação, por sua vez, deve realizar: a análise dos valores recebidos, a tradução desses valores para comandos compatíveis com os instrumentos utilizados e as leituras solicitadas pelo usuário. No caso de algumas dessas leituras é necessária a geração de figuras adequadas que representem as formas de onda ou valores presentes em alguns dos equipamentos.

---

[7] VISA — *Virtual Instrument Software Architecture* — consiste em uma interface padronizada para o controle de instrumentos através da qual a interface utilizada pelo equipamento, VXI, GPIB, PXI ou serial, pode ser abstraida pelo desenvolvedor.

[8] IVI — *Interchangeable Virtual Instruments* — Padrão estabelecido através de um consórcio entre fabricantes de equipamentos, integradores de sistema e usuários finais com o objetivo de estender o conceito de *driver* de instrumento incorporando, entre outras características, independência de hardware e capacidade de simulação [26].

A aplicação CGI recebe do formulário preenchido pelo usuário as informações apresentadas na tabela abaixo, necessárias à correta configuração dos equipamentos de medida.

Multímetro	multimetro_02
acoplamento_canal_01	1
escala_vertical_canal_01	1
unidade_escala_vertical_canal_01	1E+0
offset_canal_01	0
unidade_offset_canal_01	1E-3
Canal_de_Osciloscopio	canal_osciloscopio_02
acoplamento_canal_02	0
escala_vertical_canal_02	500
unidade_escala_vertical_canal_02	1E-3
offset_canal_02	0
unidade_offset_canal_02	1E-3
Canal_de_Osciloscopio	canal_osciloscopio_03
acoplamento_canal_03	0
escala_vertical_canal_03	50
unidade_escala_vertical_canal_03	1E-3
offset_canal_03	0
unidade_offset_canal_03	1E-3
acoplamento_canal_04	1
escala_vertical_canal_04	1
unidade_escala_vertical_canal_04	1E+0
offset_canal_04	0
unidade_offset_canal_04	1E-3
base_tempo	200µs/div
trigger_source	2
trigger_slope	Rising Edge
trigger_level_value	25
trigger_level_unit	1E-3
Gerador_Sinal_Onda	5
Gerador_Sinal_Frequencia	
Gerador_Sinal_Unidade_Frequencia	1E+0
Gerador_Sinal_Amplitude	
Gerador_Sinal_Unidade_Amplitude	1E-3
Gerador_Sinal_Offset	
Gerador_Sinal_Unidade_Offset	1E-3
Fonte_DC_Tensao_Canal_01	
Fonte_DC_Unidade_Tensao_Canal_01	mV
Fonte_DC_Tensao_Canal_02	10
Fonte_DC_Unidade_Tensao_Canal_02	V
Corrente_da_Fonte	fonte_canal_02

Tabela 7 - Variáveis fornecidas pelo formulário de medidas.

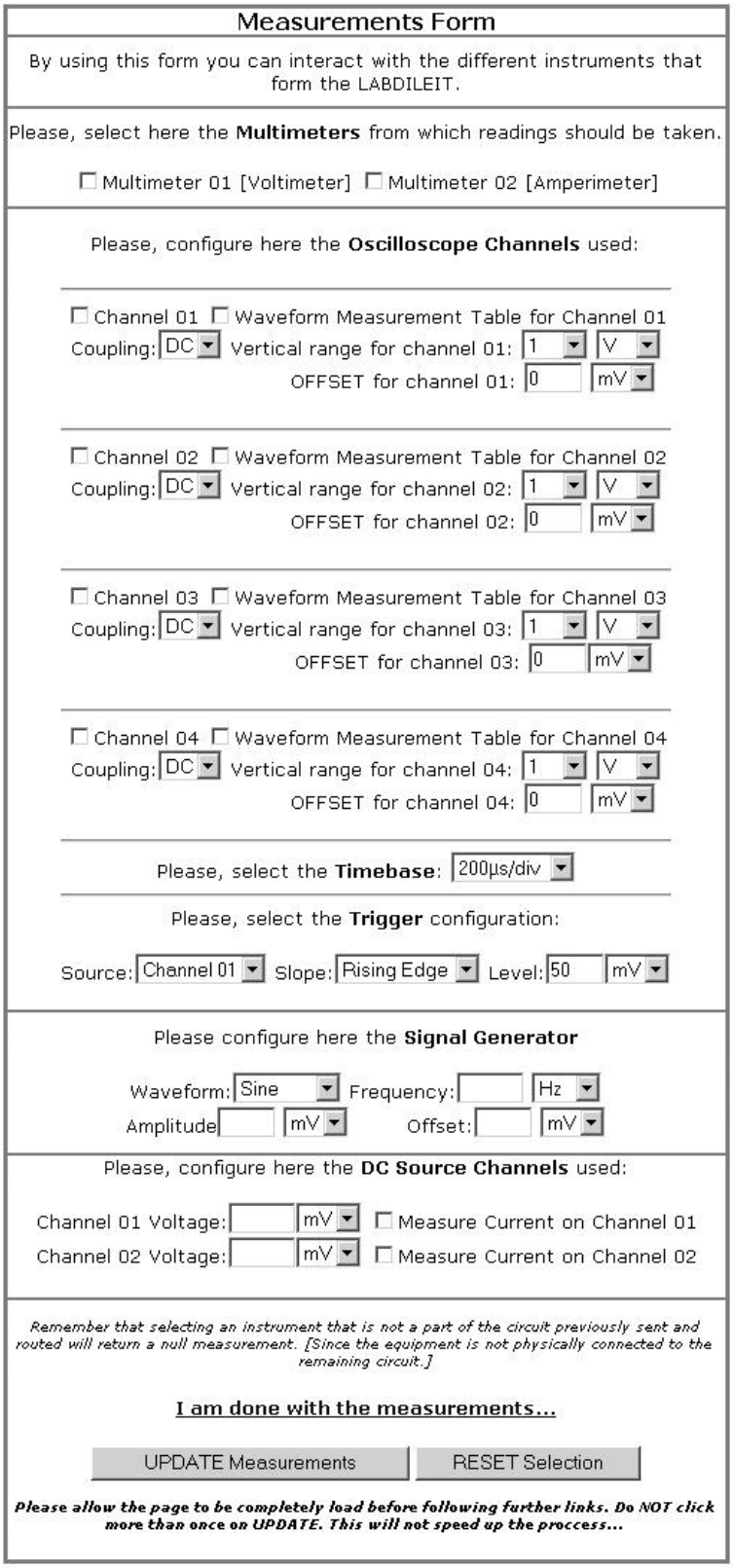
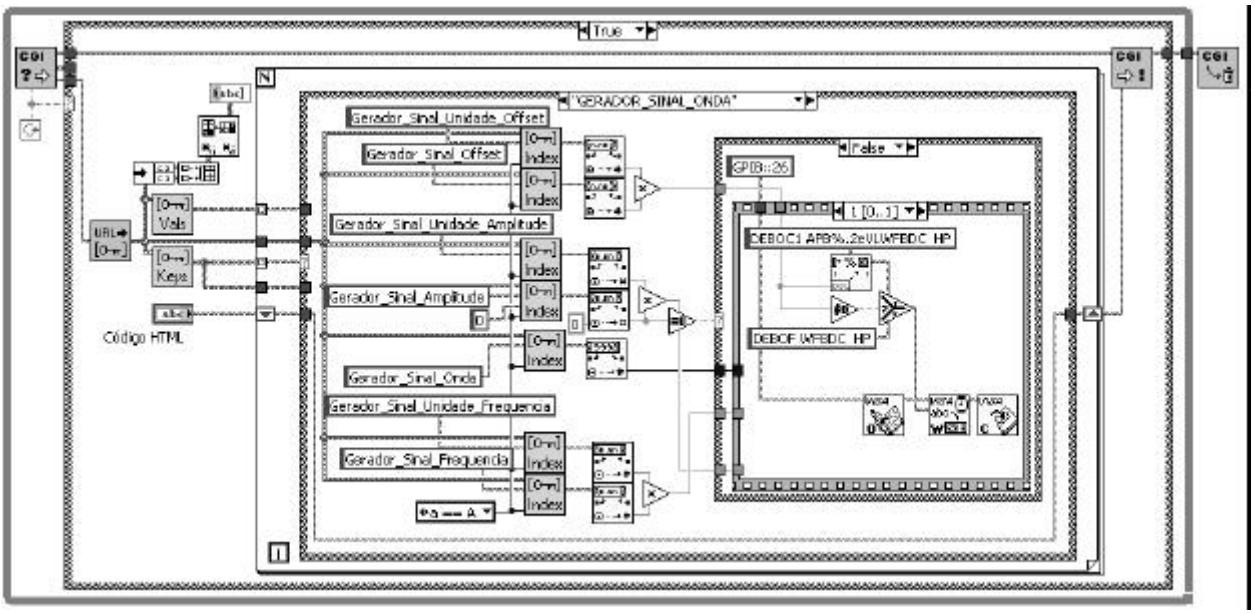


Figura 14 - Formulário de medidas.

A aplicação CGI responsável pelo tratamento das informações recebidas possui a estrutura representada pelo código 11. Nesse código é possível identificar a estrutura de um seqüenciador, que percorre todas as variáveis recebidas, configurando cada instrumento de acordo com as informações enviadas. Esse código também é o responsável pela preparação do novo formulário de medidas ao usuário, que deve agora, além dos controles originalmente disponíveis, conter também os resultados da interação anterior do usuário. Dessa forma é criado o laço de interação entre o usuário e os equipamentos de medida.



Código 11 - Diagrama responsável pelas configurações do gerador de sinais.

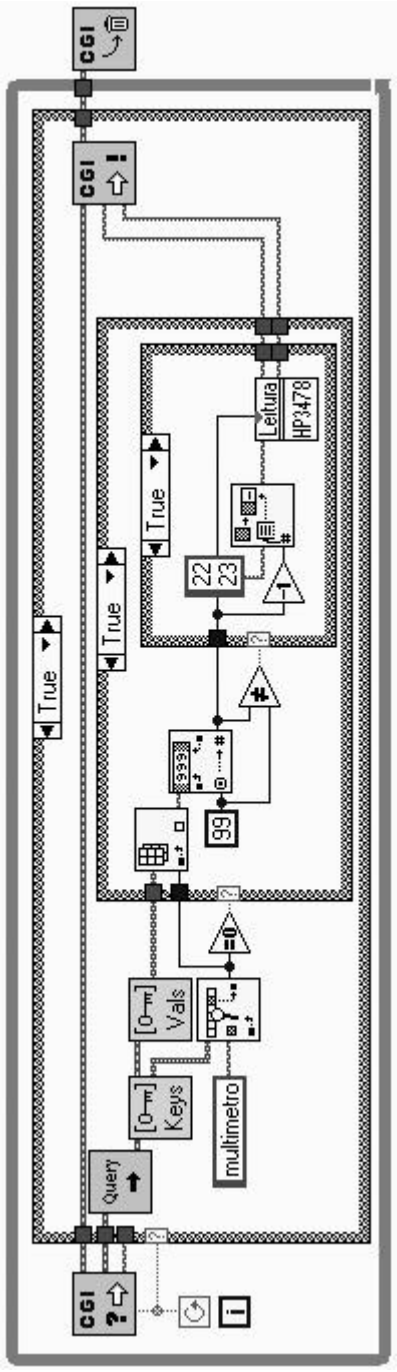
A inserção dos resultados, tais como aquisição das telas do osciloscópio, dos multímetros e da corrente fornecida pela fonte é realizada através da inserção de *links* para outros aplicativos CGI no código HTML produzido como resposta. Dessa maneira, no momento do carregamento pelo navegador da página de resposta produzida, outros aplicativos CGI serão disparados; sendo cada um deles responsável pela captura e geração de uma das várias imagens que compõem o novo formulário de medidas disponível.

Por exemplo, a leitura de um dos multímetros é realizada pelo *link* exibido no quadro abaixo, inserido no documento enviado como resposta ao usuário:

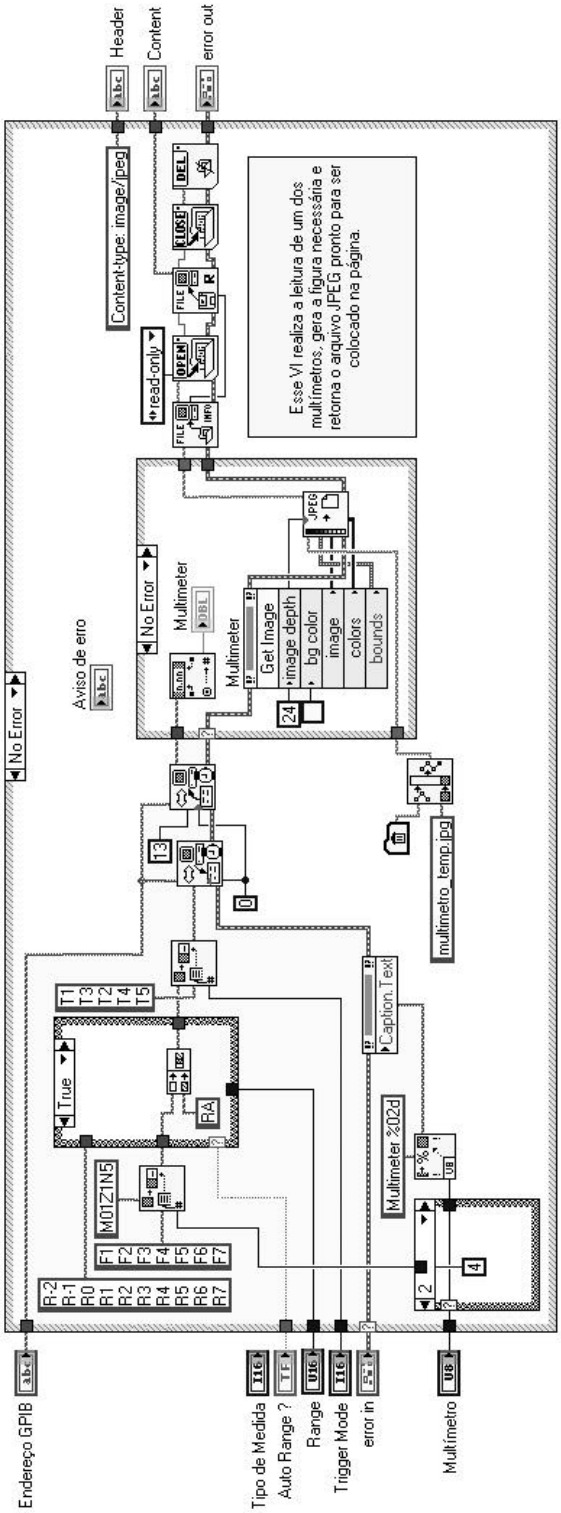
```
<img src=”http://143.106.8.162:100/cgi-bin/multimetro.vi?multimetro=01”>
```

Esse *link* leva o navegador do usuário a acionar o aplicativo *multímetro.vi*, passando para esse aplicativo a variável *multímetro* com o valor *1*. Nesse aplicativo, reproduzido pelo código 12, os argumentos passados à aplicação são inicialmente verificados, e o endereço GPIB do multímetro selecionado é então passado à aplicação cujo diagrama encontra-se retratado através do código 13.

Nesse aplicativo os comandos GPIB são adequadamente construídos e enviados ao equipamento. Em seguida o valor proveniente da leitura do equipamento é exibido em um indicador e através da aplicação a este do método *Get Image* é produzida uma imagem cujo conteúdo será retornado ao navegador do usuário. Um aspecto importante quando aplicações CGI retornam imagens é o valor passado ao navegador do cliente como cabeçalho do documento retornado. No caso particular de imagens do tipo JPEG, como as utilizadas nesse trabalho, o navegador deve ser advertido a respeito de como interpretar os dados retornados através do cabeçalho *Content-type: image/jpeg*.



Código 12 - Aplicativo para leitura do multímetro.



Código 13 - Aplicativo para geração da imagem do painel do instrumento.

Dessa mesma maneira descrita anteriormente são realizadas as inserções das imagens relativas a cada um dos canais do osciloscópio utilizados, bem como dos indicadores relacionados às fontes de polarização, conforme ilustram as figuras apresentadas abaixo.

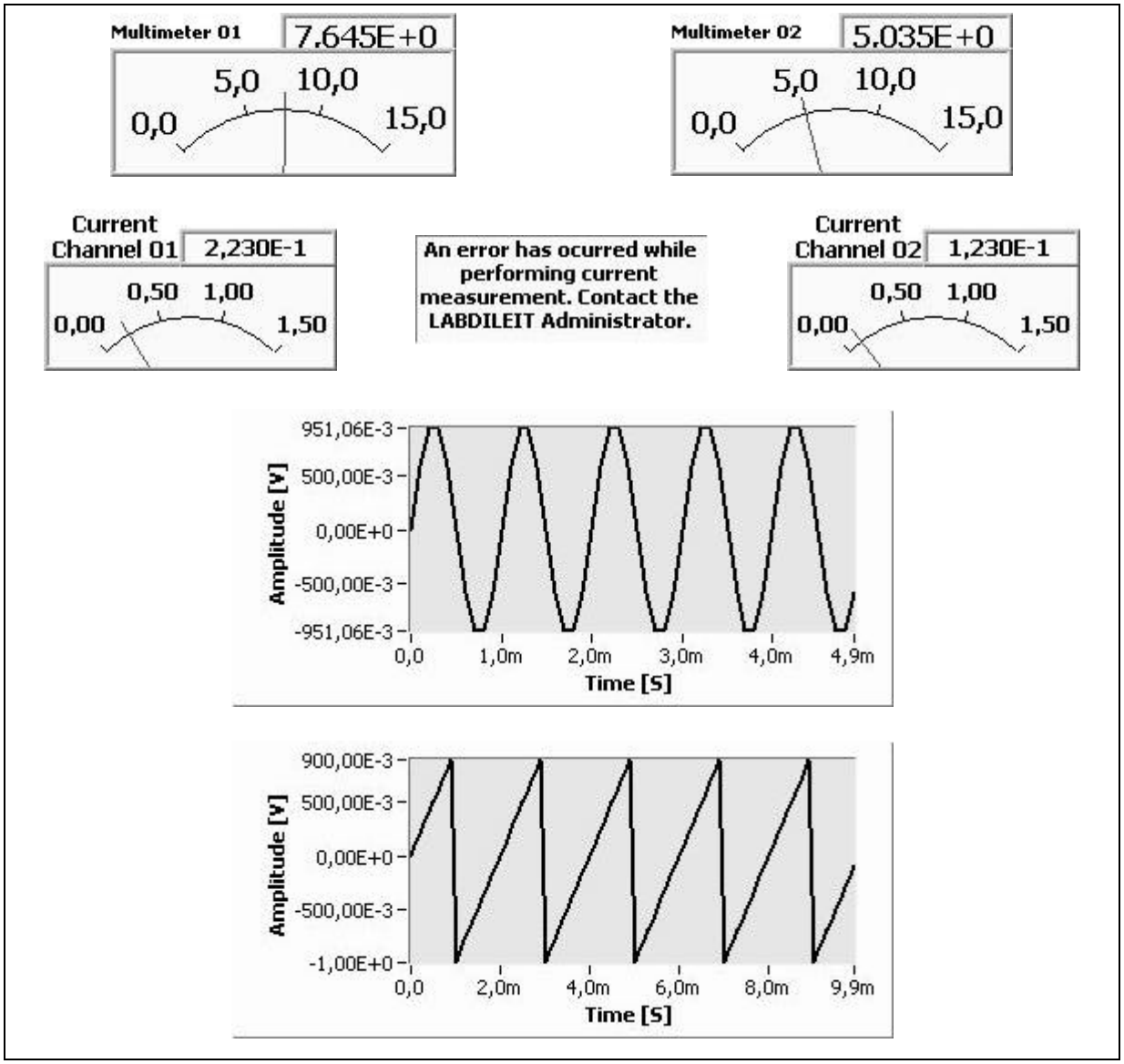


Figura 15 - Exemplos de figuras geradas dinamicamente.

Ao final do formulário de medidas exibido ao usuário como resultado de sua primeira interação com os equipamentos é inserido um *link* que permite ao usuário encerrar sua utilização

dos recursos remotos. Através desse *link* o aplicativo CGI *release_all.vi* é acionado e todos os equipamentos conectados ao barramento GPIB são reinicializados. Na matriz de interconexão todos os relés são abertos.

Capítulo 3 - A Matriz de Interconexão

A criação de mecanismos que confiram à um aglutinado de componentes a habilidade de serem interligados de forma que funções mais complexas possam ser implementadas sempre foi alvo de pesquisas e estudos, que resultaram no desenvolvimento das FPGAs — *Field Programmable Gate Arrays* — e FPAAs — *Field Programmable Analog Arrays* —, dispositivos onde elementos digitais e analógicos, respectivamente, podem ser unidos de diferentes formas [27].

No laboratório remoto descrito, onde uma das propostas de destaque é a disponibilização apenas de componentes ou blocos avulsos ao usuário, um elemento capaz de propiciar a interconexão desses componentes e dos instrumentos de medição é fundamental.

Dentre as principais características desejáveis em uma matriz de interconexão, para a finalidade proposta, podem ser destacadas: uma arquitetura flexível; um razoável número de terminais disponíveis e, por fim, contatos elétricos eficientes.

A topologia adotada para a matriz de interconexão faz uso da estrutura de interconexão ilustrada na figura 16(a) abaixo.

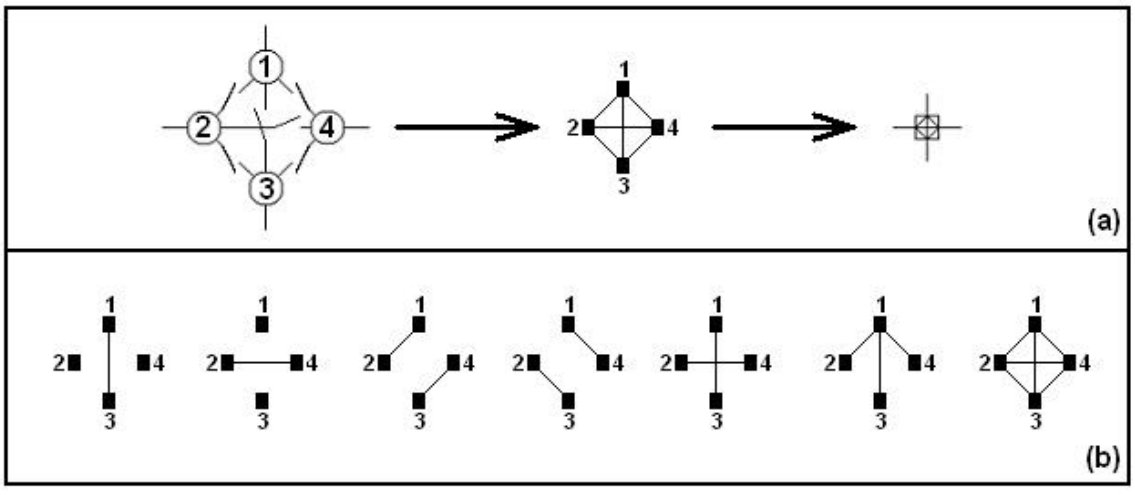


Figura 16 - (a) Estrutura dos nós complexos. (b) Algumas possíveis interconexões.

Essa estrutura, formada por seis chaves dispostas nos lados e diagonais de um quadrado, é chamada de hipernó pelo fato de permitir a conexão de um dado vértice a qualquer um dos outros três vértices [28][29]. Na figura 16(b) são representadas algumas das conexões possíveis com a estrutura adotada.

A estrutura de interconexão adotada na implementação do laboratório remoto é uma matriz quadrada de ordem 10, contendo 100 destes hipernós. Seu diagrama simplificado é mostrado na figura 17.

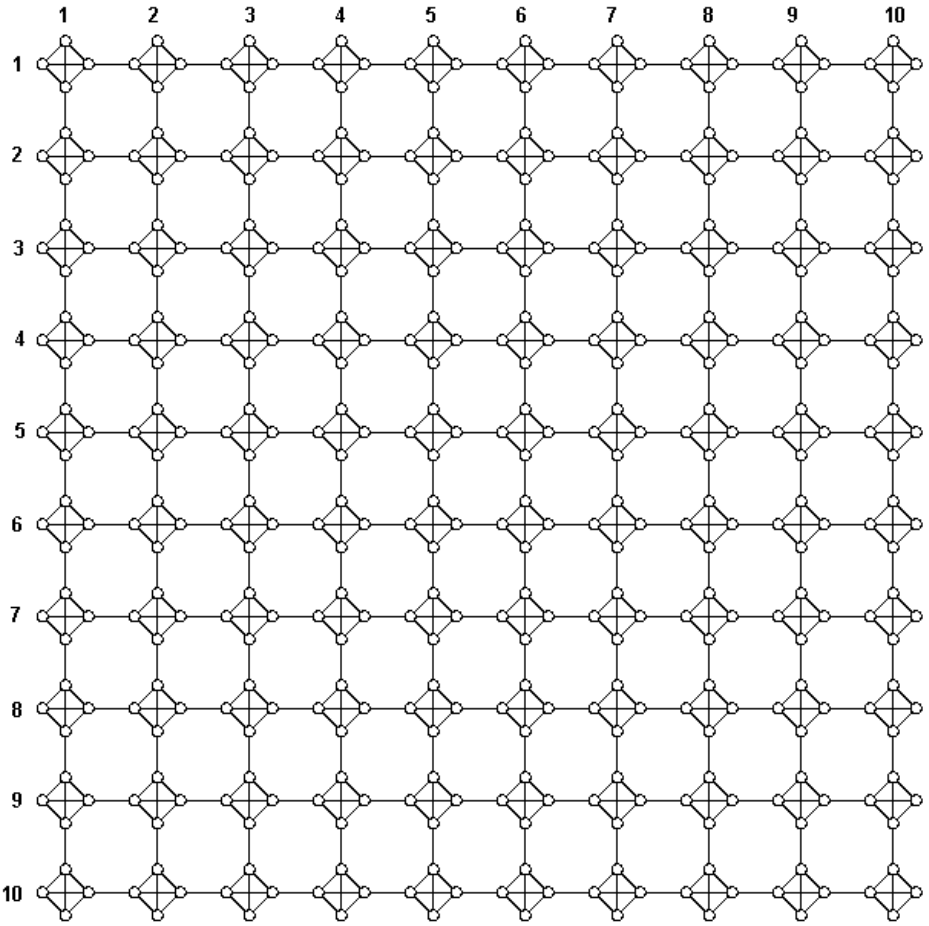


Figura 17 - Representação da matriz de interconexão.

Nesta matriz de interconexão, os quarenta terminais de borda estão ligados aos componentes e instrumentos disponíveis para o usuário.

Nesta aplicação a qualidade do contato elétrico é de importância crucial. Por esta

razão, foram adotados relés do tipo *micro-switch* com contato de baixa resistência. Dentre os vários modelos de relés disponíveis no mercado foi escolhido o relé do tipo *reed* fabricado pela indústria Metaltex [30], modelo SH1NAC-5V, tendo em vista principalmente sua baixa resistência de contato, reduzidas dimensões, vide figura 18, e disponibilidade no mercado.

A tabela 8 apresenta as demais características do relé utilizado.

Tensão nominal	5	V
Resistência bobina	500 ±10%	Ω
Tensão operação	≤ 3.8	V
Tensão desoperação	≥ 0.5	V
Potência de comutação (máx.)	10	W
Tensão de comutação (máx.)	200	VCC
Corrente de comutação (máx.)	0.5	A
Corrente de condução (max.)	1.0	A
Resistência inicial de contato (max.)	0.15	Ω
Tensão de ruptura entre contatos abertos	250	VCC
Tensão de ruptura contato-bobina	1500	VCC
Resistência de isolação entre contatos abertos	10^{10}	Ω
Resistência de isolação contato-bobina	10^{10}	Ω
Tempo de operação (máx.)	1.0	mS
Tempo de desoperação (máx.)	0.5	mS
Vida elétrica (20VCC – O.5A)	10^{5}	operações

Tabela 8 - Especificações técnicas do relé SH1NAC-5V.

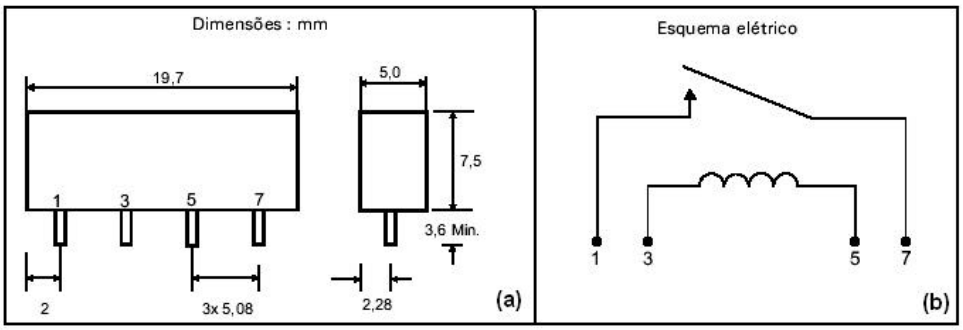


Figura 18 - Reed-relé utilizado. (a) Dimensões. (b) Esquema elétrico.

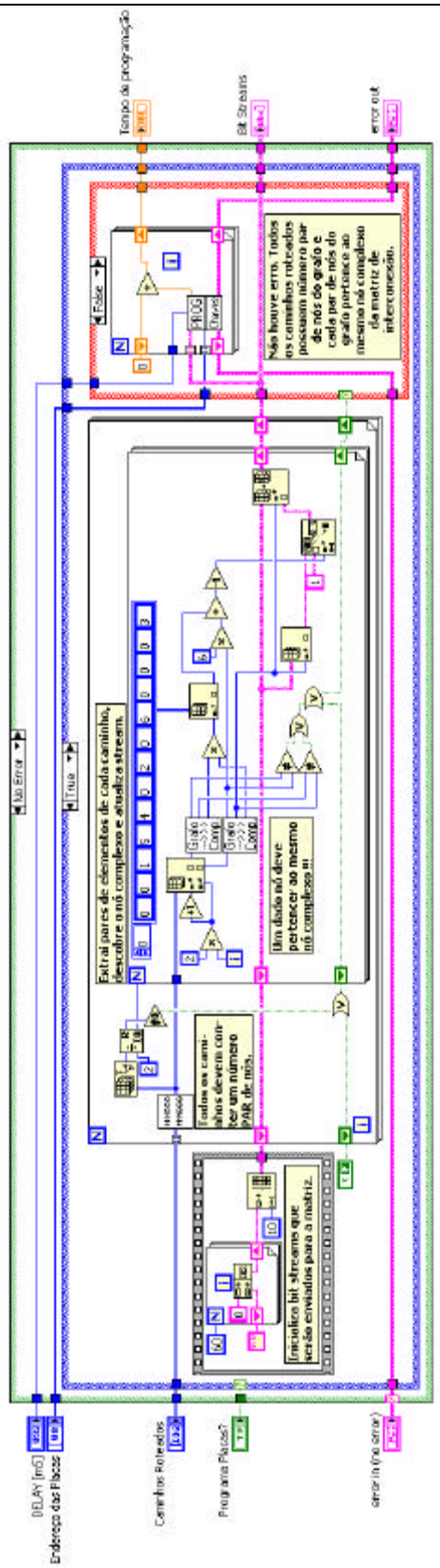
A matriz de interconexão foi dividida em 10 partes idênticas, representando cada uma das linhas da figura 17, sendo cada uma dessas partes montada em uma placa de circuito impresso separada. A simetria das estruturas facilitou sua divisão. Ao final desse trabalho, no apêndice B encontram-se todos os esquemáticos produzidos, juntamente com o projeto da placa de circuito impresso utilizada para a implementação de uma linha da matriz de interconexão.

Cada uma das placas montadas abriga 60 chaves, arranjadas de forma a comporem os 10 hipernós relativos a cada linha da matriz. Todas as placas possuem estrutura exatamente igual, diferindo apenas o endereço atribuído à cada uma delas. Cada uma das placas possui um regulador de tensão (7805), um *led* indicador de operação e um decodificador de endereços implementado através de um comparador de 4 bits (7485). Além desses componentes, há também 8 registradores de deslocamento de 8 bits (74164) e 8 circuitos integrados para o acionamento dos relés (ULN2003).

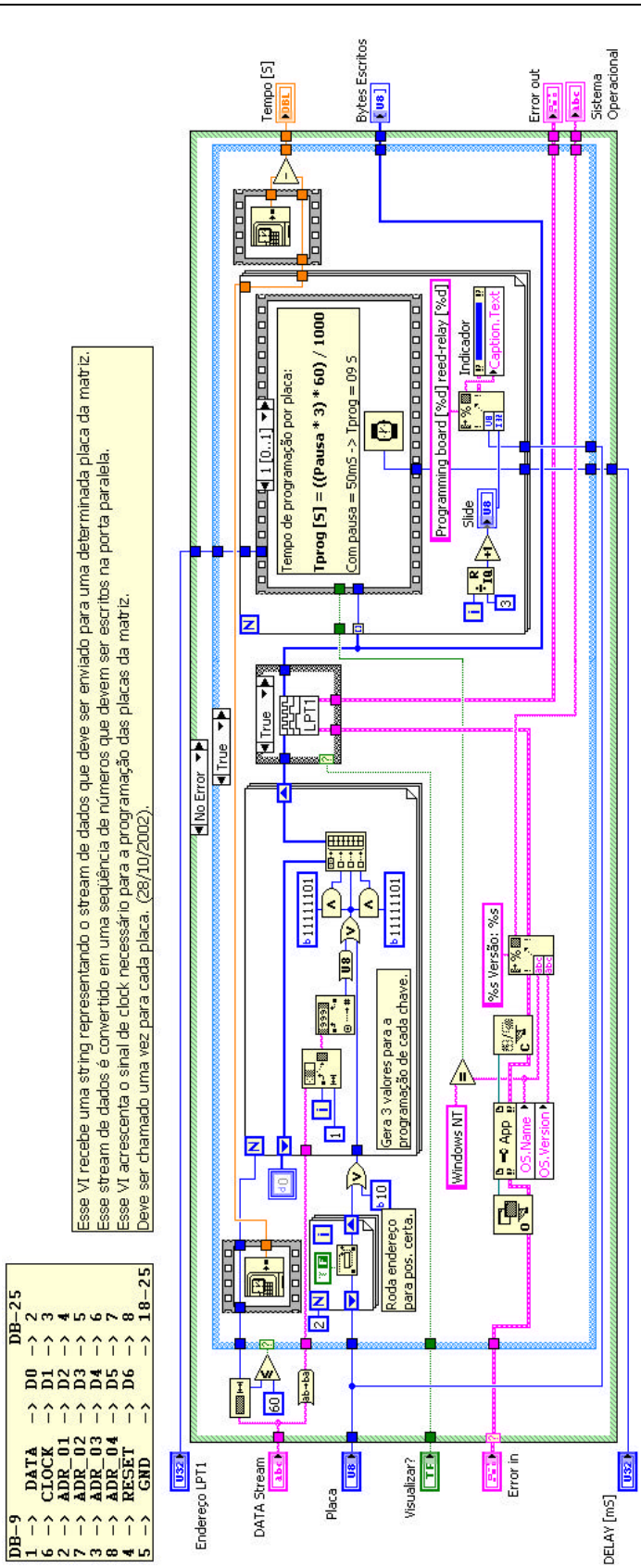
A programação das chaves que compõem a matriz ocorre através da porta paralela do microcomputador servidor do sistema, da qual 7 bits são utilizados para implementar os seguintes sinais: *clock, data, reset, address1, address2, address3* e *addres4*. Inicialmente é colocado nas 4 linhas de endereçamento um valor que selecione uma das placas para programação. Esse endereço deve permanecer até que toda a seqüência de 60 bits associados à cada uma das chaves seja carregada no registrador de deslocamento através dos sinais *clock* e *data*. O sinal de *reset* é utilizado para a liberação de todas as chaves.

Os códigos 14 e 15, na página seguinte, trazem os diagramas dos dois trechos responsáveis pela programação da matriz. O código 14 recebe informação a respeito de quais chaves devem ser fechadas e a processa de forma a verificar a consistência dos valores passados. No caso de todas as chaves serem válidas é produzido, ainda pelo código 14, os dez *bit streams* que serão posteriormente enviados, pelo código 15, à cada uma das placas que formam a matriz.

O diagrama do código 15 recebe os *bit streams* produzidos pelo diagrama do código 14 e produz cada um do números inteiros de 8 bits que deverão ser escritos na porta paralela. Esses valores devem conter as informações relativas ao endereço da placa sofrendo a programação, bem como os valores corretos para o sinal de *reset*. Após a geração desses valores acontece a escrita na porta paralela que, por depender de código diferenciado para os sistemas operacionais Windows NT/2000 e Windows 95/98, exige a identificação do sistema operacional e a decisão por diferentes fragmentos de código. Esse módulo também recebe como entradas as configurações relativas ao endereço da porta paralela e o intervalo de tempo entre a escrita de cada *byte* (o que afeta diretamente a velocidade de programação de cada placa).



Código 14 - Diagrama para conversão dos caminhos roteados.



Código 15 - Diagrama para programação da matriz.

3.1. Roteamento da Matriz

A topologia estabelecida para a matriz de interconexão exige a elaboração de um mecanismo que permita a eficiente decisão de quais chaves devem conduzir para o correto estabelecimento dos caminhos elétricos entre os terminais dos componentes situados em sua borda.

Para a determinação dessas ligações, técnicas para o cálculo de caminhos mínimos entre terminais periféricos da matriz são aplicadas de forma que, ao final do processo de roteamento, os diferentes nós elétricos que constituem um dado circuito tenham sido fisicamente interconectados.

Algoritmos para o cálculo de caminhos mínimos constituem um assunto amplamente abordado na literatura e que ainda vem sendo alvo de pesquisas que objetivam, entre outros aspectos, o desenvolvimento de algoritmos de maior eficiência e menor custo computacional. Problemas envolvendo a determinação de caminhos mínimos em grafos possuem as mais diferentes aplicações como, por exemplo, em reconhecimento de padrões, problemas de inteligência artificial, jogos matemáticos e de labirintos, etc.[31][32].

Um grafo $G=(X,U)$ pode ser definido através de dois vetores denominados X e U. O primeiro vetor contém os vértices ou nós do grafo, enquanto o segundo descreve, através de pares ordenados, os arcos desse grafo. A quantidade N de vértices de um grafo define sua ordem. No caso de grafos orientados, também conhecidos como dígrafos, os elementos i e j de cada arco $u=(i,j)$ indicam, respectivamente, o vértice de origem e de destino daquele arco. Em muitas aplicações, incluindo o processo de roteamento descrito neste trabalho, a orientação dos arcos é irrelevante, importando apenas a existência de conexões entre os vértices do grafo.

A cada um dos arcos $u \hat{I} U$ de um dado grafo G podem ser associados valores $l(u) \hat{I} \hat{A}$, representando o comprimento daquele arco e, cujo significado real esteja intimamente relacionado com o problema em questão, podendo ser interpretados como o custo do deslocamento ou de construção de um caminho ligando um vértice à outro, tempos de deslocamento ou, como é o caso neste trabalho, a possibilidade ou não da utilização de determinada chave na matriz de interconexão. Com o objetivo de facilitar a notação, o custo associado à $u(i,j)$ é representado na forma L_{ij}, ou ainda L_j quando se trata do custo de um nó

definido como origem até um dado nó *j*.

O cálculo do caminho mínimo ligando um vértice de origem, denominado 1, até todos os outros vértices do grafo é realizado através da aplicação do algoritmo de Moore e Djikstra, referido muitas vezes como algoritmo de Djikstra apenas [32]. Este algoritmo é um dos mais difundidos e conhecidos para o cálculo de caminhos mínimos, sendo particularmente adequado para situações onde o custo ou comprimento dos arcos é positivo. Princípio de Bellman, ilustrado no quadro abaixo é a base para o algoritmo empregado.

Princípio de Bellman: *Se P:1® j é um caminho mínimo de 1 até j em um grafo G e (i,j) é o último arco de P, então P_i® i é um caminho mínimo de 1® i.*

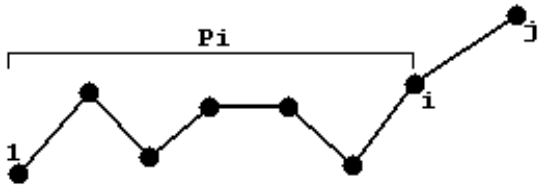
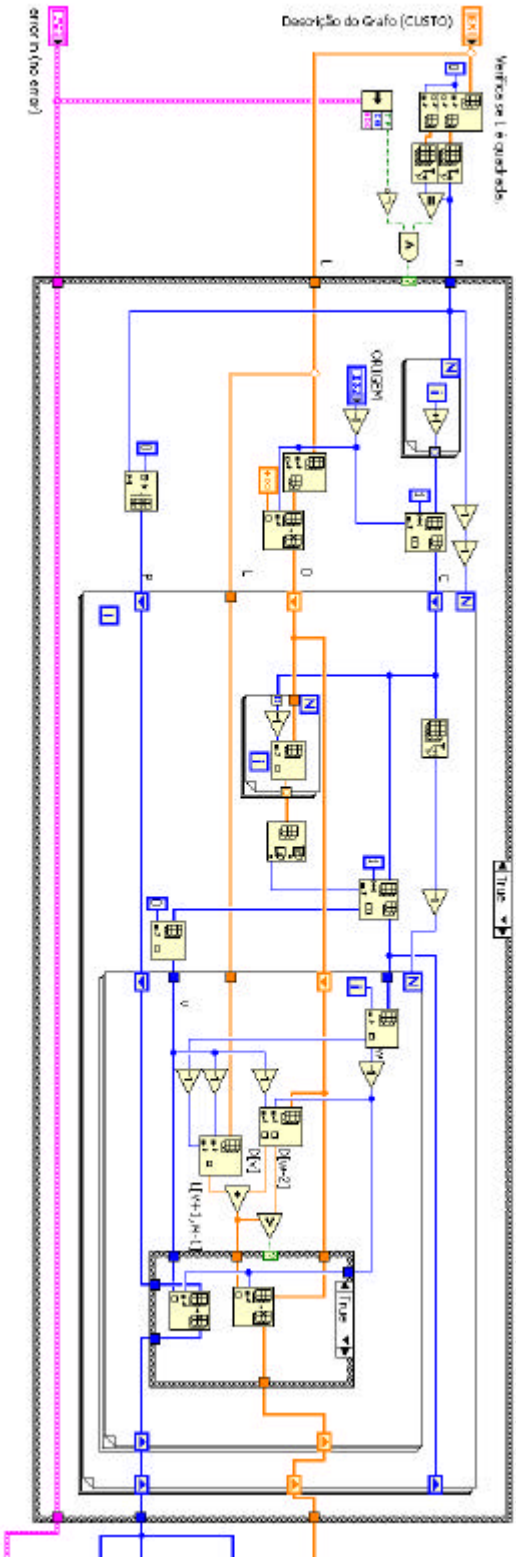


Figura 19 - Princípio de Bellman.

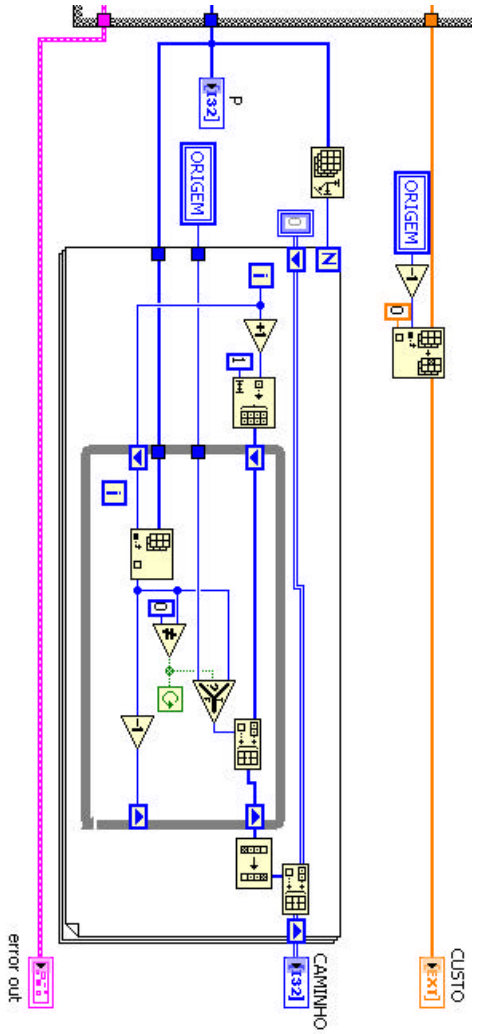
Dessa forma, para um *j* fixo podem ser obtidos vários caminhos *1® j* tomando os caminhos mais curtos P_i até vários *i* para os quais existe no grafo *G* um arcos de ligação *(i,j)* e somá-los aos caminho P_i. Esses caminhos possuem comprimentos $L_i + L_{ij}$; onde L_i é o comprimento de P_i. É possível agora, tomar o mínimo sobre *i*, isto é, escolher *i* tal que a soma $L_i + L_{ij}$ seja a menor possível. De acordo com o princípio de Bellman, este é o caminho mais curto *1® j* e o que possui o comprimento dado pelas chamadas equações de Bellman.

$$L_1 = 0$$
$$L_j = \min_{i \neq j}(L_i + L_{ij}); \quad \text{para } j = 2,...,n$$

O algoritmo de Dijkstra é na realidade um procedimento de marcação de vértices de um dado grafo. A cada passo, um determinado vértice pode receber uma marcação permanente ou temporária, seja o caminho da origem até o vértice em questão o mais curto ou não, respectivamente.



Código 16 - Algoritmo de DIJKSTRA.



Algoritmo de DIJKSTRA (continuação)

A implementação em ambiente LabVIEW do algoritmo de roteamento foi realizada através da construção do VI cujo ícone se encontra apresentado a seguir.

Figura 20 - Implementação do algoritmo de DIJKSTRA.

A figura 17, sugere que a matriz de admitância para toda a matriz de interconexão possa ser composta pela justaposição das matrizes de admitância associadas à cada um dos hipernós, mais as ligações entre esses hipernós que, na realidade, não representam de fato chaves. O módulo de roteamento desenvolvido e descrito nas próximas seções não considera algumas das características de simetria, pois foi considerado que o ganho real não supera o custo de implantação.

A representação computacional do grafo associado à matriz é realizada através de uma matriz de admitância, onde cada elemento representa o custo entre os nós representados pelas colunas e linhas da matriz. O fato do grafo associado à matriz de interconexão ser não orientado, a torna simétrica. Isto, por sua vez, permite economia em seu armazenamento e tratamento.

As entradas para esse VI são respectivamente o nó do grafo considerado como origem e a matriz de descrição do grafo, além da estrutura de erro. Após sua execução são fornecidas como saída uma matriz contendo em cada uma de suas linhas o caminho que leva do nó definido como origem a cada um dos outros nós do grafo, além de um vetor contendo o custo de cada um desses caminhos. Também, a estrutura de erro é fornecida na saída. O diagrama responsável pela implementação do algoritmo de roteamento é exibido pelo código 16.

Observando o diagrama da figura 17, é possível identificar a verificação das dimensões da matriz de custo para assegurar que se trata de uma matriz quadrada. Não havendo nenhum erro, seja ele recebido externamente ou gerado internamente, pelo fato da matriz *L* não ser quadrada, os vetores *C*, *D*, e *P* são inicializados e o software inicia as repetições que efetivamente implementam as iterações que compõem o algoritmo de Dijkstra. Ao final dessas iterações os vetores *P* e *D* estão preenchidos. Entretanto, *P* ainda não contém a descrição explícita dos caminhos ligando a origem aos demais nós do grafo. Essa tarefa é realizada pelas repetições finais que transformam o vetor *P* na matriz de caminhos.

A tarefa de interligar os terminais periféricos da matriz com vistas à formação de circuitos elétricos não é, porém, resolvida apenas com a aplicação do algoritmo de roteamento, pois esse define apenas um caminho entre dois terminais de componentes. Para que um circuito seja de fato constituído são necessárias as interligações de vários terminais e, para tanto, a cada roteamento bem sucedido a descrição da matriz deve ser atualizada para refletir que agora alguns

caminhos já não mais podem ser utilizados, pois formam outro nó elétrico do circuito. Além disso, alguns mecanismos necessitam ser introduzidos para que o processo de roteamento desfrute de alguma flexibilidade nos casos onde o roteamento de alguns caminhos não sejam imediatamente bem sucedidos.

O fato de a matriz de admitância sofrer constantes alterações após cada roteamento, eleva a probabilidade de situações onde um dado caminho não possa ser estabelecido. Nesses casos, o software marca a situação que levou à essa impossibilidade e reinicializa o processo de roteamento não daquele caminho, mas sim de todo o circuito. Dessa forma eleva-se a probabilidade de sucesso ao final do processo.

Outro aspecto bastante interessante está relacionado aos nós elétricos formados pela junção de apenas dois terminais da matriz ou pela junção de mais de dois desses terminais. No primeiro caso, o cálculo do menor caminho entre os terminais envolvidos leva à melhor solução para o problema. Já no último caso, a solução adotada foi a escolha de dois terminais inicias e, após a determinação do melhor caminho entre eles, as distâncias entre o próximo terminal da matriz e todos os pontos que pertencem ao caminho previamente calculado são determinadas. Esse processo se repete até que todos os terminais que compõem um nó elétrico tenham sido conectados.

O *layout* das placas de circuito impresso que formam a matriz de interconexão, apresentados no apêndice B, comporta todos os pontos de interconexão entre as placas em um dos lados da placa de circuito impresso. Na representação da figura 17 essas conexões representam as conexões entre as linhas do grafo associado. A facilidade de manutenção do hardware conseguida com o deslocamento desses pontos de ligação é ponderada com um aumento no comprimento das interconexões físicas estabelecidas pela matriz. Esse fato pode, em parte, ser compensado com a introdução no grafo associado de um gradiente de valores para os custos de cada chave. Dessa forma, cada chave pode passar a ter um custo tanto mais alto quanto mais à esquerda estiver situada no grafo da figura 17. O estabelecimento de gradientes de custo permite uma melhora no comprimento do caminhos roteados e sua utilização pode ser expandida a cada nó elétrico corretamente estabelecido, permitindo que regiões mais densamente utilizadas da matriz possam ser evitadas.

Todas as funções discutidas nos parágrafos anteriores, relativas ao problema de

roteamento e programação das chaves que formam a matriz foram reunidas no módulo exibido na figura 21 abaixo. A interface ilustrada na figura não contém informações relevantes ao usuário remoto. Entretanto é de grande valia para a aferição do roteamento e na análise da influência de vários parâmetros no resultado final do roteamento.

Através desse módulo são definidos parâmetros fundamentais para o funcionamento do sistema, tais como a definição dos componentes atualmente ligados ao terminais periféricos da matriz de interconexão e o número máximo de tentativas de roteamento. Também através desse módulo de roteamento características estéticas da figura de roteamento produzido e disponibilizada para o usuário remoto podem ser configuradas. Na parte superior direita da interface apresentada são exibidas informações que também podem ser relevantes, tais como a quantidade de chaves utilizadas no total, para a maior e para a menor conexão realizada, o tempo de roteamento e o tempo de programação da matriz.

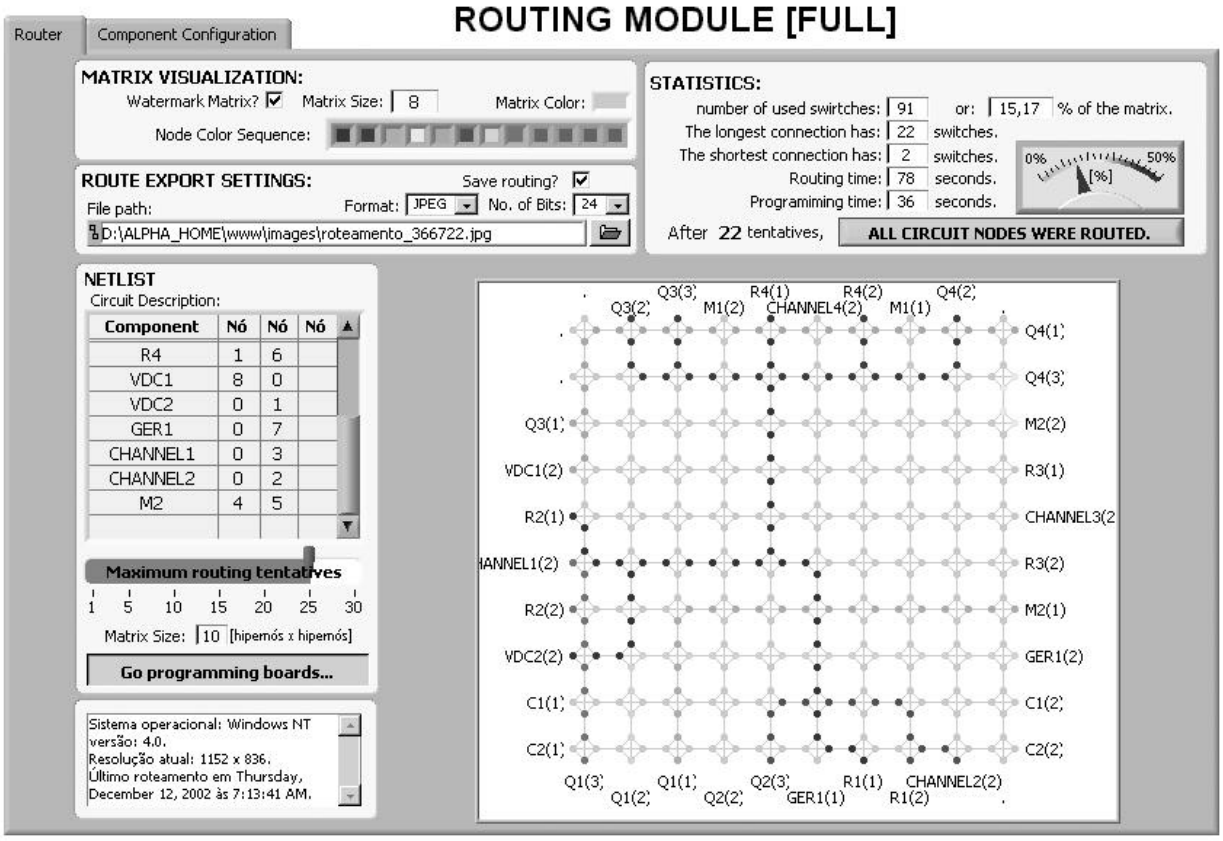


Figura 21 - Módulo de roteamento. Vista da interface.

A esse módulo está vinculada grande parte do código que forma o sistema e que por razões de espaço não pode ser apresentada em maior detalhe no decorrer do texto. A figura 22 ilustra as funções desenvolvidas e utilizadas pelo módulo apresentado na figura 21.

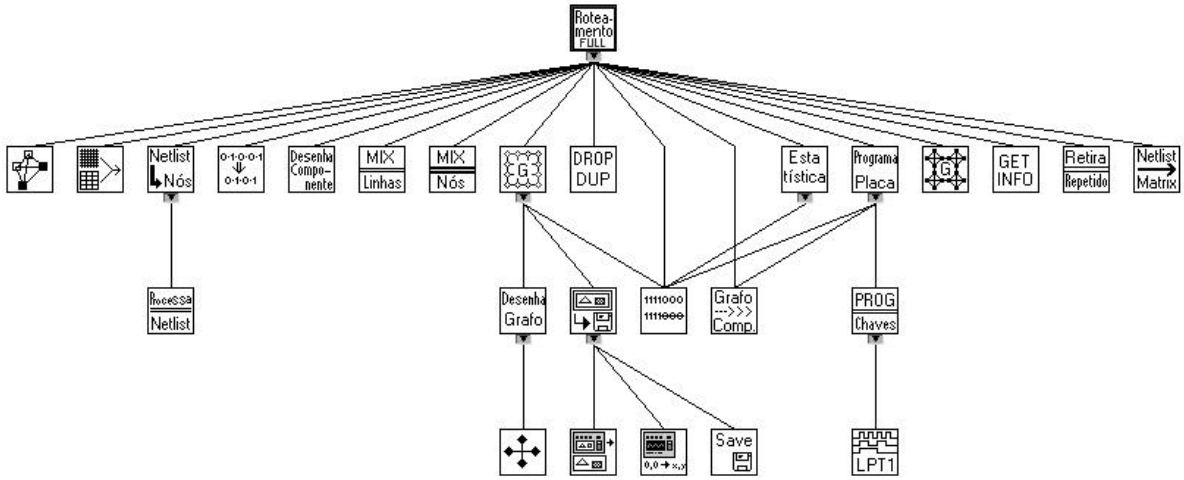


Figura 22 - Hierarquia de funções do módulo de roteamento.

Rotea-mento	Módulo completo de roteamento.	Retira Repetido	Remove nós repetidos.
	Implementação do algoritmo de roteamento de Djikstra.	Netlist Matrix	Define os terminais da matriz que devem ser interconectados.
	Altera os valores de custo das chaves do grafo associado à matriz.	Processa Netlist	Analisa o *netlist* recebido.
Netlist Nós	Realiza o mapeamento entre os terminais dos componentes e os termina is da matriz.	Desenha Grafo	Desenha o grafo associado.
	Retira elementos repetidos de um vetor.		Grava o conteúdo do controle *picture* para um arquivo.
Desenha Compo-nente	Desenha os nomes dos componentes na borda do grafo.	1111000 1111000	Retira os zeros do final de um vetor.
MIX Linhas	Embaralha as linhas de uma matriz de entrada.	Grafo Comp.	Determina à qual nó complexo pertence um dado nó do grafo.
MIX Nós	Embaralha os elementos de um vetor.	PROG Chaves	Gera o *bit stream* que será enviado pela porta paralela até a matriz.
	Desenha os nós do grafo.		Desenha um dado nó complexo.

	Filtra elementos repetidos para desenho no grafo.		Extrai o conteúdo de um controle *picture*.
	Produz as informações à respeito do número de chaves utilizadas, etc...		Extrai a origem e as dimensões do controle *picture*.
	Realiza a programação das placas da matriz.		Grava a imagem em um arquivo.
	Sintetiza a matriz de admitância associada à matriz de interconexão.		Realiza as operações de escrita pela porta paralela.
	Recolhe informações sobre o sistema operacional, etc...		

Tabela 9 - Breve descrição das funções que compõem o módulo de roteamento.

Capítulo 4 - Resultados Experimentais

Neste capítulo são apresentados os resultados experimentais obtidos com testes feitos para alguns circuitos utilizando-se a matriz descrita no capítulo anterior como elemento de interconexão. Os circuitos testados fazem parte de experimentos normalmente adotados em disciplinas experimentais de eletrônica.

Para cada um dos circuitos apresentados são mostrados o resultado do roteamento, o *netlist* enviado ao sistema e algumas formas de onda.

4.1. A Configuração da Matriz de Interconexão

Durante o período de obtenção dos resultados apresentados neste capítulo, a matriz de interconexão foi configurada de acordo com a figura abaixo.

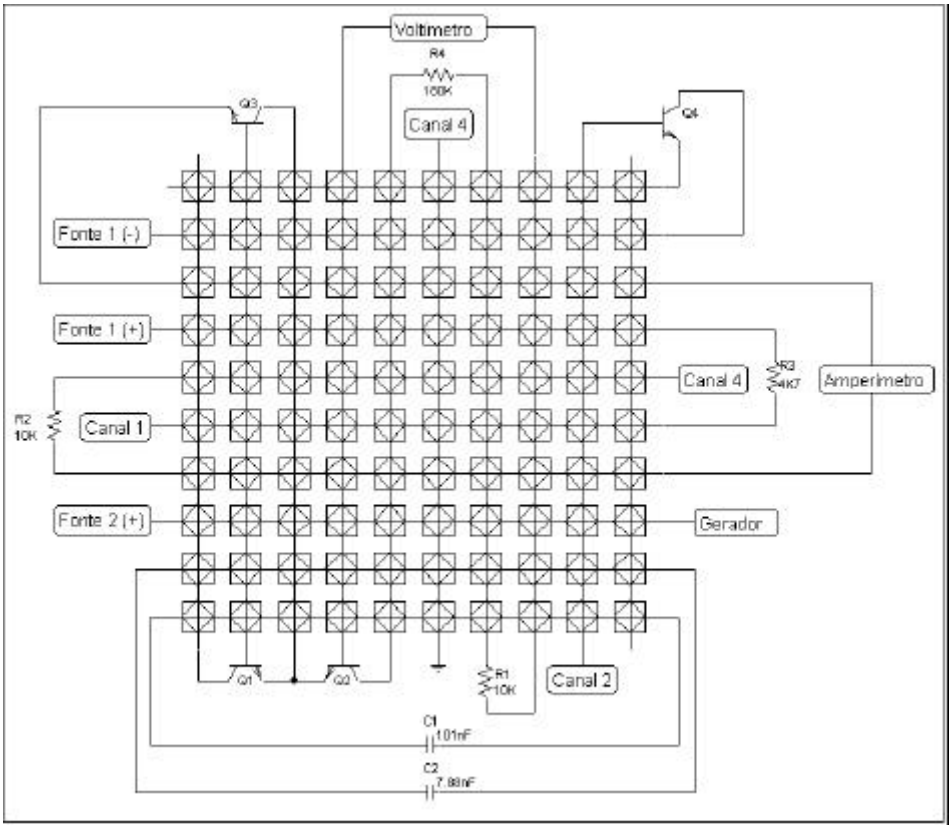


Figura 23 - Matriz de interconexão durante o levantamento dos resultados.

Através da figura 23 é possível identificar os componentes presentes na matriz de interconexão e, portanto, disponíveis ao usuário durante a construção de seu circuito. A informação de quais componentes estão atualmente disponíveis é fornecida ao usuário assim que esse efetua com sucesso seu *login* no sistema.

A tabela 10 resume as associações utilizadas entre os nomes utilizados na figura 23 e aqueles utilizados no decorrer desse capítulo nos resultados dos roteamentos obtidos.

Componente	Sigla	Valores
Fontes de polarização	VDC1, VDC2	0 – 15V Ajustável
Resistores	R1, R2, R3, R4	10K, 10K, 4K7, 180K
Capacitores	C1, C2	101nF, 7.88nF
Transistores	Q1, Q2, Q3, Q4	C.I. LM3046
Canais de osciloscópio	CHANNEL 1, CHANNEL 2, CHANNEL 3, CHANNEL 4,	Osciloscópio HP54503
Amperímetro	M2	
Voltímetro	M1	
Gerador de Sinais	GER1	

Tabela 10 - Componentes presentes na matriz.

Os transistores utilizados são provenientes do circuito integrado LM3046, que possui no total 5 transistores bipolares interconectados conforme sugerido abaixo.

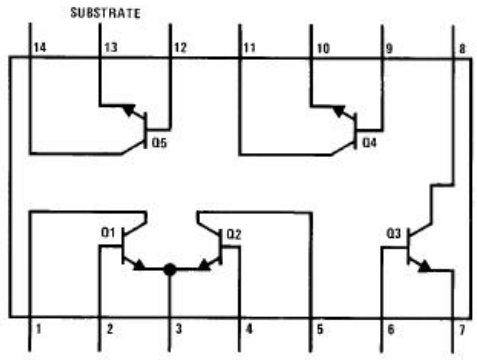


Figura 24 - Esquemático do circuito integrado LM3046.

Para que a descrição do circuito enviada ao sistema seja corretamente interpretada é importante que os transistores utilizados tenham seus terminais descritos segundo a ordem:

emissor, base e coletor.

Na figura 23, apenas um dos terminais referentes aos canais de osciloscópio, ao gerador de sinal e à uma das fontes de polarização aparecem na matriz de interconexão. A razão para isso é que todos os canais de osciloscópio, bem como um dos terminais do gerador de sinal e da fonte de polarização estão permanentemente unidos e conectados ao ponto considerado como de terra da matriz. Este ponto comum, que aparece sinalizado nas figuras de roteamento como GER1(1), deve ser referido durante a descrição do circuito como sendo o nó 0.

O número total de terminais utilizados na matriz de interconexão no momento de geração dos resultados apresentados foi de 36 terminais. Os quatro terminais não utilizados aparecem nas figuras de roteamento sinalizados através de um ponto.

4.2. Circuito RC Série

O circuito formado pela associação série de um resistor e um capacitor, apresentado na figura 25, é um dos primeiros circuitos elétricos abordados nos cursos de engenharia. Através desse circuito é possível o estudo de alguns conceitos como, por exemplo, a carga e a descarga exponencial de um capacitor, sua resposta AC e sua resposta em freqüência.

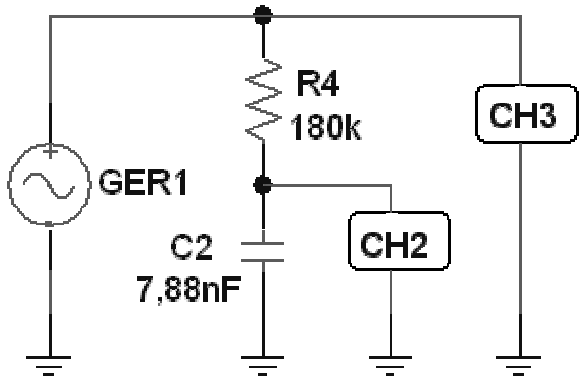


Figura 25 - Circuito RC.

Netlist

```
GER1    0     1
R4      1     2
C2      0     2
CHANNEL2    0     2
CHANNEL3    0     1
```

Ao circuito RC tal como descrito na figura 25, corresponde o roteamento exibido na figura 26. A situação de interconexão conseguida para um dado circuito é sempre enviada de

volta ao usuário, pois permite à este melhor avaliar o impacto que a matriz de interconexão pode inserir nos resultados.

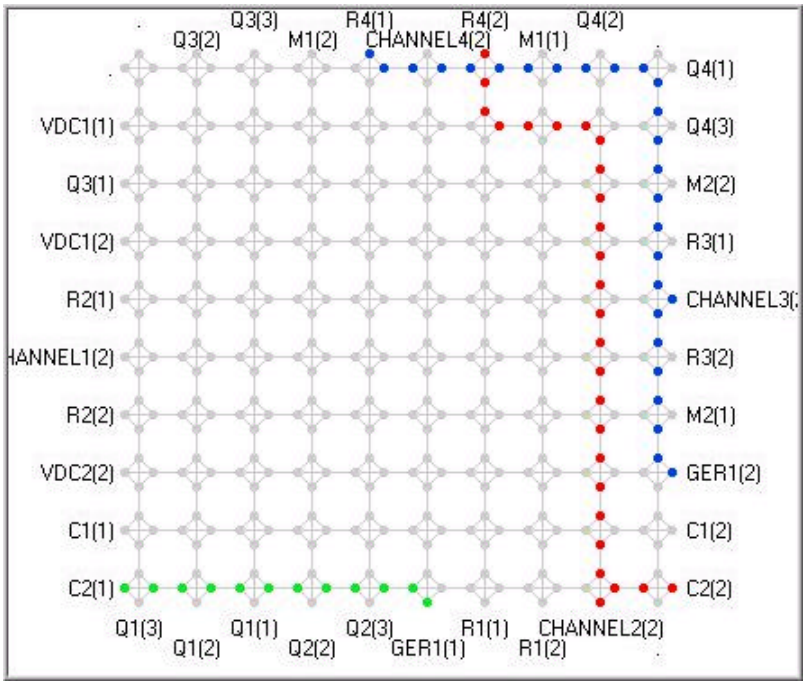


Figura 26 - Roteamento para o circuito RC.

A figura 26 permite identificar as conexões entre um dos terminais do capacitor e a ponta de prova do osciloscópio, e entre o outro terminal do capacitor e o ponto de terra como sendo ligações curtas quando comparadas com as demais. Vale lembrar que interconexões entre as linhas do grafo associado à matriz de interconexão são mais longas por que as conexões entre as placas de circuito impresso são mantidas em uma das bordas da placa.

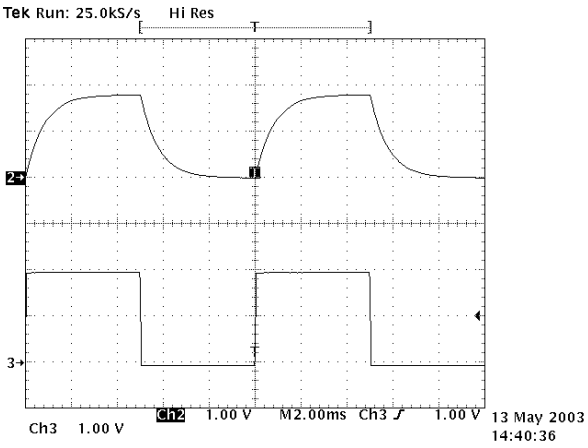


Figura 27 - Circuito RC: carga e descarga.

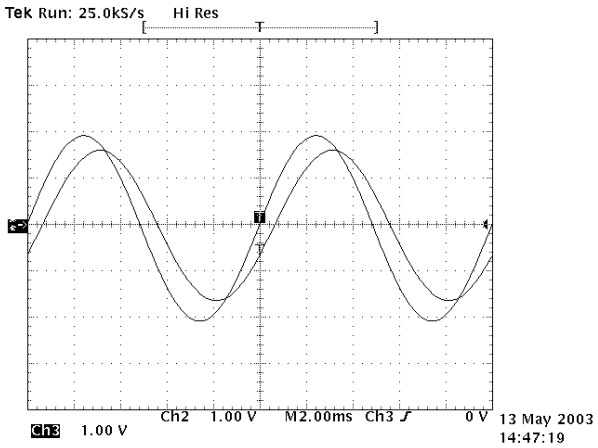


Figura 28 - Circuito RC: Excitação senoidal.

Inicialmente, foi aplicado ao circuito RC uma onda quadrada variando de 0 a 2 Volts e de freqüência igual a 1kHz. Dessa forma é possível observar o processo de carga e descarga exponencial do circuito, conforme mostra a figura 27.

O mesmo circuito foi submetido a uma entrada senoidal de amplitude 2V, sem *offset* e de freqüência novamente igual a 1kHz. A figura 28 traz os resultados, onde é possível observar o atraso de fase introduzido pelo capacitor bem como a atenuação imposta ao sinal.

4.3. Restaurador DC

O estudo de dispositivos semicondutores nos cursos de eletrônica é iniciado invariavelmente através da apresentação e caracterização de diodos. Dentre os vários circuitos comumente abordados o restaurador de nível DC apresentado na figura 29, foi escolhido como próximo exemplo.

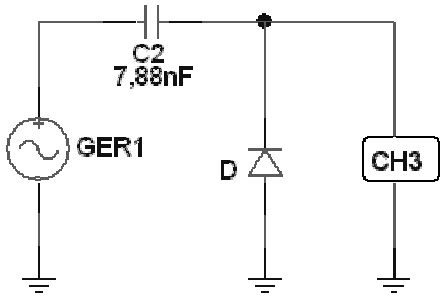


Figura 29 - Restaurador DC com diodo.

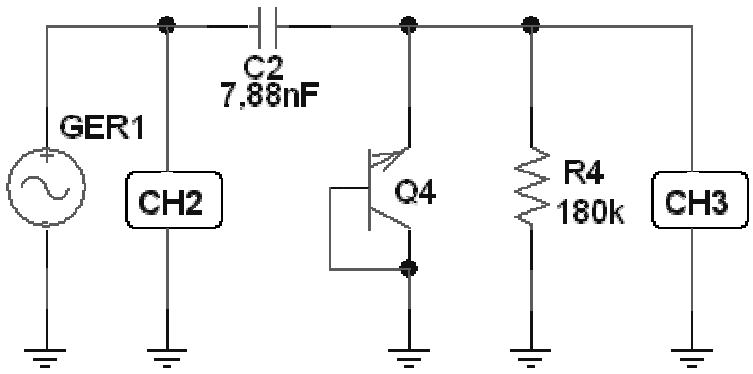


Figura 30 - Restaurador DC com carga.

Netlist

```
GER1       0     1
CHANNEL2   0     1
C2         1     2
(R4        0     2)
Q4         2     0     0
CHANNEL3   0     2
```

O princípio de operação do circuito consiste em se armazenar no capacitor um valor de tensão igual a parte negativa da tensão de entrada. Essa tensão, somada à parte positiva do sinal de entrada, faz com que esse seja deslocado e passe a apresentar apenas valores positivos. De forma análoga, caso o diodo tenha sua polaridade invertida, o capacitor armazenará tensão correspondente à parte positiva do sinal de entrada, e como resultado o sinal de saída ficará restrito a valores negativos apenas.

Pelo fato de não existirem diodos na matriz de interconexão um dos transistores do circuito integrado LM3046 teve sua base e seu coletor unidos através da própria matriz para que apenas uma de suas junções fosse utilizada.

Na primeira descrição enviada para o laboratório remoto, o restaurador DC foi montado sem o resistor de carga e, por essa razão, o *netlist* enviado teve a linha que descreve a interconexão de R4 suprimida.

No segundo experimento envolvendo esse circuito, foi adicionado à sua descrição um resistor de carga R4, no valor de 180k.

As figuras 31 e 32 exibem os resultados do roteamento conseguidos para ambas as situações.

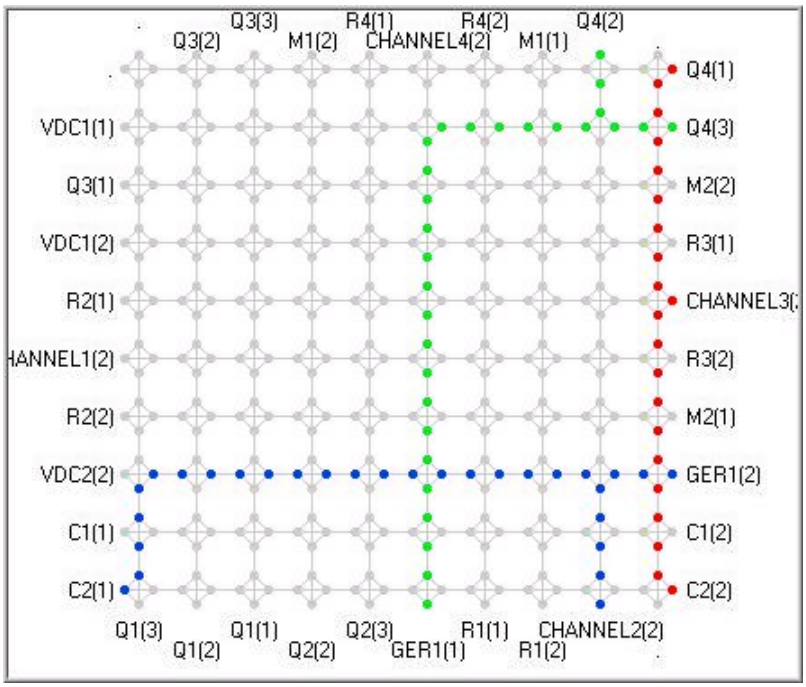


Figura 31 - Roteamento para o restaurador DC sem carga.

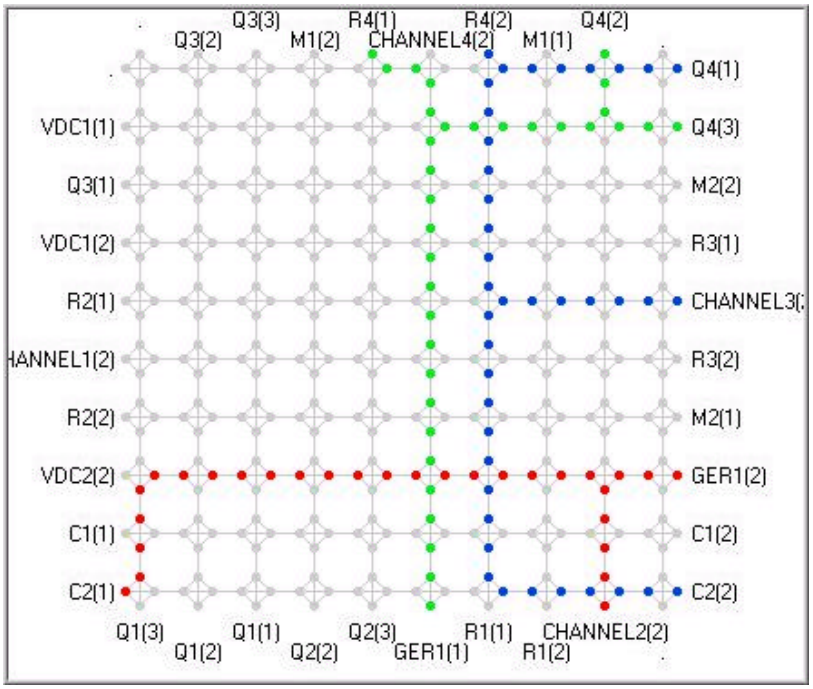


Figura 32 - Roteamento para o restaurador DC com carga.

Na situação sem carga o circuito foi estimulado com uma onda quadrada variando de –2 a 0 Volts e de freqüência igual a 10kHz. Nessa situação a figura 33 ilustra as curvas obtidas.

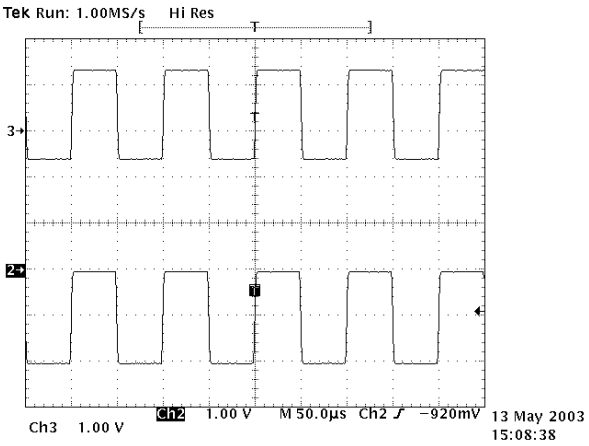


Figura 33 - Restaurador DC: sem carga.

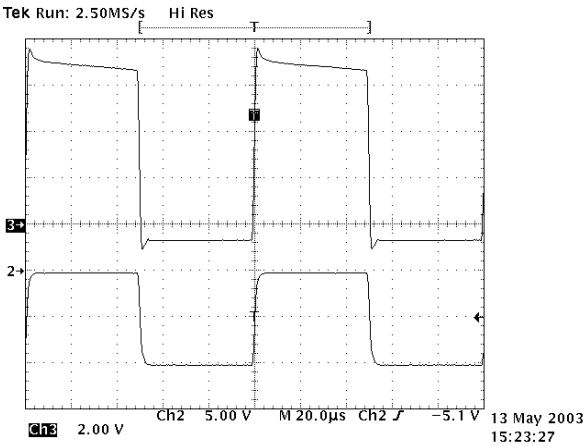


Figura 34 - Restaurador DC: com carga.

Na situação onde o resistor de 180k está presente o circuito foi estimulado com uma onda quadrada excursionando de –10 a 0 Volts e de freqüência igual a 10kHz. As curvas obtidas são mostradas na figura 34, através da qual é possível avaliar o efeito de carga e descarga do capacitor pelo resistor de carga, nos momentos em que o diodo entra e sai de condução.

4.4. Amplificador de Emissor Comum

Os amplificadores de um estágio nas configurações emissor-comum, base-comum e coletor-comum são, sem dúvida, circuitos largamente abordados nos cursos de graduação em eletrônica. Por essa razão, o amplificador de emissor comum, ilustrado na figura 35, foi escolhido para exemplificar a capacidade do laboratório remoto em permitir a montagem de circuitos utilizando transistores.

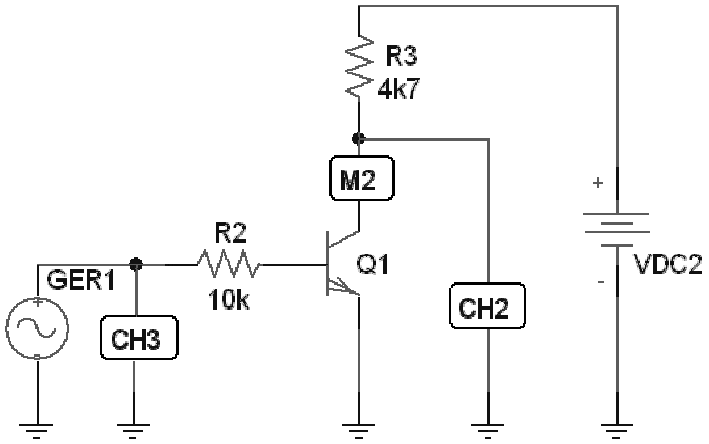


Figura 35 - Amplificador de emissor comum.

Netlist

GER1		0	1
CHANNEL3		0	1
R2		1	2
Q1	0	2	3
M2		3	4
R3		4	5
CHANNEL2		0	4
VDC2		0	5

A figura 36 ilustra o roteamento conseguido para o *netlist* do circuito da figura 33. Uma vez corretamente montado, foi aplicado à base do transistor um sinal senoidal de amplitude igual a 45mV, *offset* de 750mV e freqüência igual a 1kHz. A fonte de polarização VDC2 foi ajustada em 5 Volts. A figura 35 ilustra as formas de onda presentes no gerador de sinais e no coletor do transistor, permitindo a identificação da característica inversora do circuito juntamente com seu ganho em tensão de aproximadamente A_V = 40. Do amperímetro M2 foi realizada a leitura I_C =300µA.

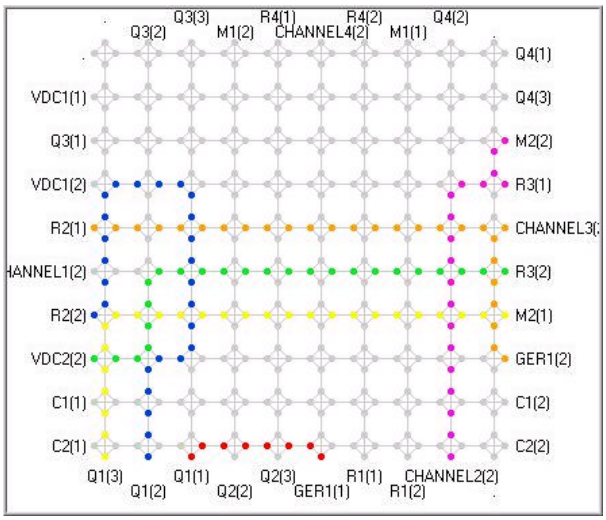


Figura 36 - Roteamento para o amplificador de emissor comum.

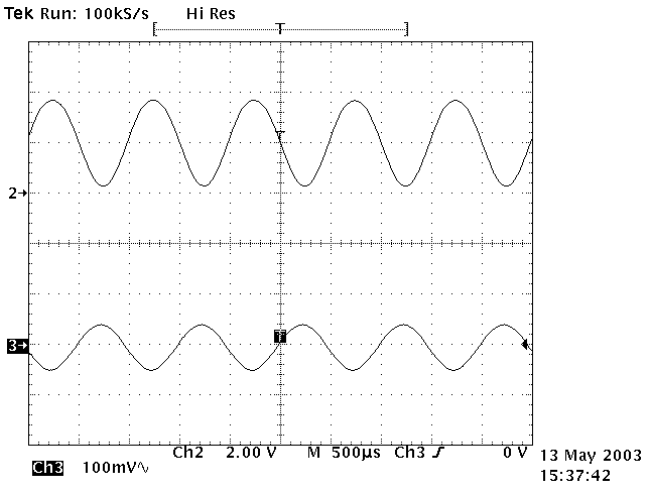


Figura 37 - Amplificador de emissor comum.

4.5. Amplificador Diferencial

O amplificador diferencial mostrado na figura 38 foi escolhido como último exemplo de aplicação do laboratório remoto tendo em vista seu maior nível de complexidade, elevado número de componentes e de nós elétricos.

O circuito da figura 38 e seu *netlist* foram utilizados para avaliar o comportamento do sistema de roteamento da matriz de interconexão, tendo em vista o maior número de chaves utilizadas da matriz.

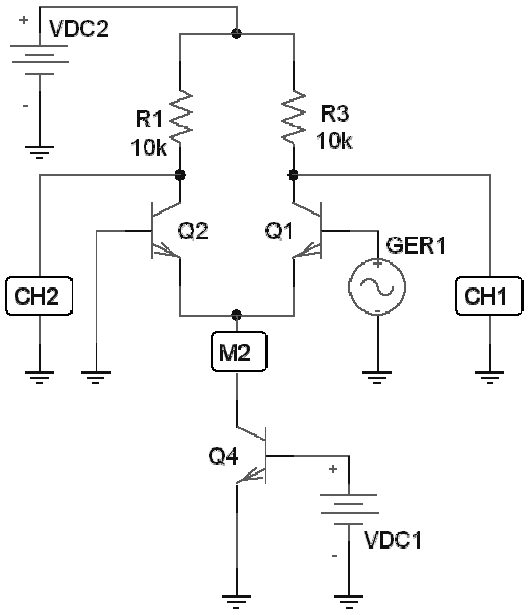


Figura 38 - Amplificador diferencial.

Netlist

Q1	4	7	3
Q2	4	0	2
Q4	0	6	5
R1		1	2
R2		1	3
VDC1		0	6
VDC2		0	1
GER1		0	7
CHANNEL1		0	2
CHANNEL2		5	4
M2		5	4

O *netlist* apresentado foi enviado seis vezes consecutivas sendo que em duas delas o sistema não conseguiu rotear todos os caminhos necessários à construção do circuito. Durante este teste, o módulo de roteamento apresentado na figura 21, estava configurado para um número máximo de 15 tentativas de roteamento. Elevando-se o número máximo de tentativas de roteamento a probabilidade de sucesso é também elevada, juntamente, entretanto, com o tempo total necessário entre o envio do circuito e o início das medidas.

É interessante mencionar que um dado circuito pode originar, a cada novo envio, um

novo roteamento. Isso pode levar, em algumas situações, a situações onde a repetibilidade de algumas medidas pode ser prejudicada, tendo em vista as diferentes características que algumas conexões podem assumir cada vez que o circuito é novamente montado. Essas variações dependem, entre outros fatores, da freqüência de operação dos sinais envolvidos no circuito analisado.

Os resultados para vários roteamentos de uma mesma topologia são apresentados na figura seguinte e ilustram os aspectos discutidos anteriormente.

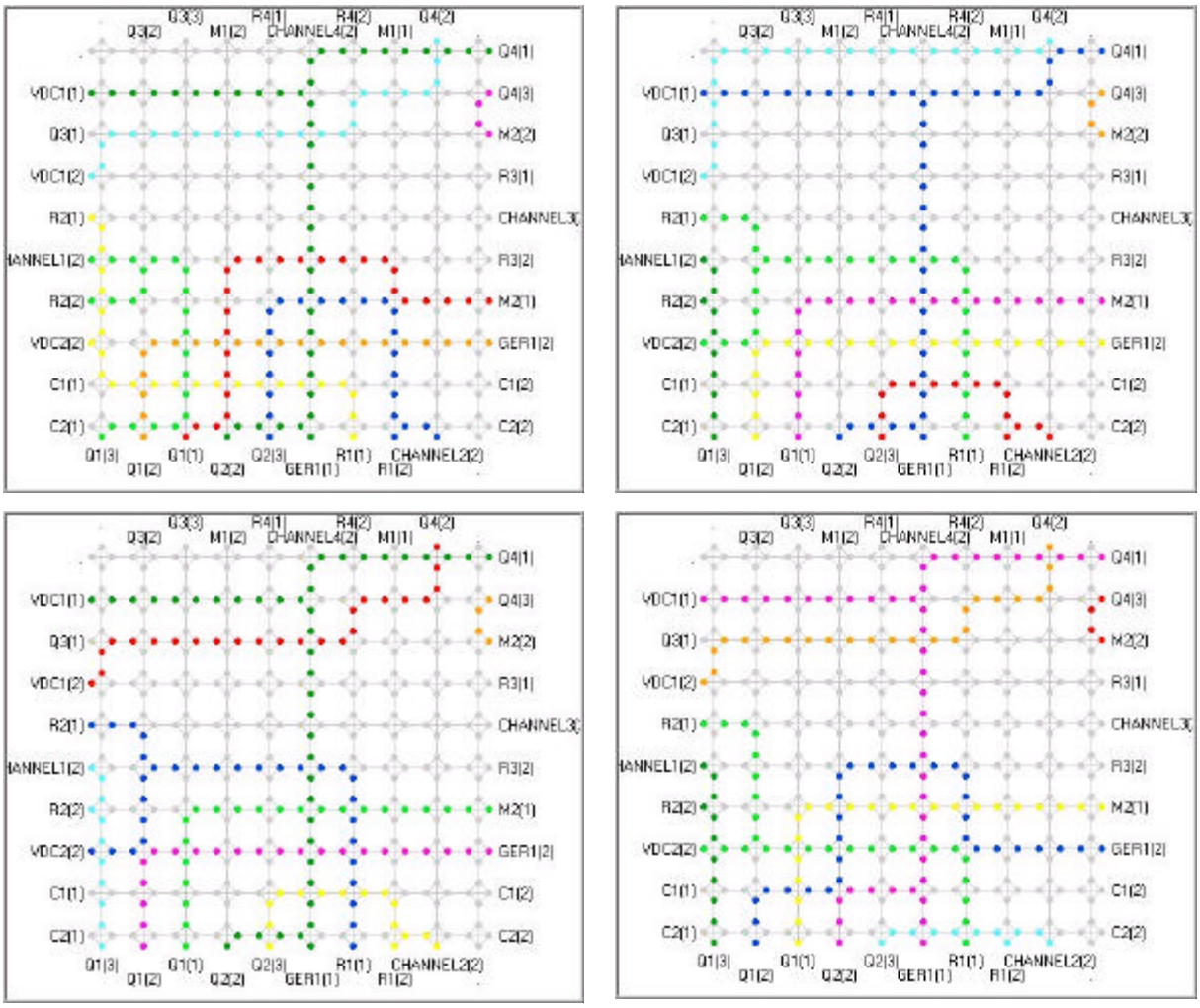


Figura 39 - Roteamentos para o amplificador diferencial.

Capítulo 5 - Conclusões

O presente trabalho apresentou a implementação de um ambiente remotamente controlado capaz de permitir a seus usuários a prototipagem, através da Internet, de uma gama de circuitos eletrônicos suficientemente variada para que o sistema encontre, dentre inúmeras, aplicação no aprendizado de eletrônica.

O sistema implementado apresenta características complementares aos trabalhos até então encontrados na literatura. Entre elas, o fato de serem disponibilizados apenas componentes eletrônicos descompromissados e instrumentos de medição e estímulo; sendo atribuição do usuário, a descrição do circuito, dos estímulos aplicados e a definição dos pontos de medida. A configuração dos equipamentos de medida também é de responsabilidade do usuário do sistema.

O presente trabalho cobriu, na plataforma de software adotada para o desenvolvimento, conhecimentos que se estendem desde a manipulação simples de vetores e matrizes para o estabelecimento dos caminhos pela matriz, até sua programação, realizada através da porta paralela do microcomputador servidor, passando pelo acesso a bancos de dados externos através das tecnologias ODBC e SQL, controle de instrumentos através da interface GPIB e implementação de interfaces com o usuário baseadas em navegadores e na Internet.

O sistema alcançado ao final desse trabalho atende aos objetivos inicialmente propostos no sentido em que permite que medidas em circuitos dinamicamente montados sejam realizadas, utilizando-se para isso como cliente um microcomputador PC equipado com um navegador padrão e uma conexão à Internet de baixa velocidade.

Capítulo 6 - Sugestões para trabalhos futuros

Considerando a complexidade inerente à um sistema de experimentação remota definitivo, o trabalho descrito deve ser encarado como um passo adicional no caminho da implantação de um sistema completo.

Nos seguintes parágrafos são abordadas algumas melhorias que podem ser exploradas em futuros trabalhos de pesquisa nesta mesma linha da experimentação remota voltada à educação nos currículos de engenharia.

A primeira delas seria a introdução de uma câmera de vídeo, por exemplo, as atualmente utilizadas para videoconferência, transmitindo imagens em tempo real dos painéis de alguns dos equipamentos utilizados, melhorando assim a interação do usuário com alguns dos equipamentos, principalmente equipamentos como osciloscópios.

A introdução de um banco de circuitos comumente utilizados, ou recomendados para análise, pode poupar os esforços dedicados ao processo de roteamento da matriz. Como vantagem adicional pode elevar a repetibilidade das medidas, visto que os caminhos utilizados pela matriz permanecem constantes. Esse recurso pode se tornar particularmente atraente durante a utilização do sistema proposto como ferramenta adicional no acompanhamento de aulas de laboratório durante um determinado semestre, onde a ordem e a topologia dos circuitos sob análise costumam estar bem definidas.

Ainda em relação ao processo de roteamento da matriz de interconexão, mecanismos que permitam ao usuário definir manualmente os caminhos utilizados para pontos críticos de seu circuito, ou mesmo para todas as interconexões existentes, podem ser oferecidos como ferramentas avançadas. Tais mecanismos podem ser de grande valia para que os usuários identifiquem as influências que as conexões realizadas através da matriz podem estar introduzindo no circuito sob análise.

A integração de módulos de simulação do lado do servidor, num primeiro momento apenas do ponto de operação do circuito enviado, também pode enriquecer o sistema no sentido em que pode prevenir de forma mais eficiente ligações que tendam a danificar certos componentes ou mesmo partes da matriz de interconexão.

A natureza *on-line* do sistema descrito introduz alguns problemas relacionados ao

fato de que todos os equipamentos envolvidos, incluindo a própria matriz de interconexão, devem permanecer energizados durante todo o tempo. A introdução de um módulo, composto de hardware e software, capaz de energizar e desenergizar os equipamentos todos de forma racional representa um importante recurso para que a disponibilização do sistema ocorra de fato durante os maiores períodos de tempo possíveis.

A quantidade relativamente grande de chaves presentes na matriz de interconexão introduz a possibilidade de desenvolvimento de um módulo, também composto de hardware e software, capaz de aplicar à matriz uma seqüência de testes do forma a verificar o funcionamento de todas as suas partes. A realimentação necessária à conclusão sobre o funcionamento das partes envolvidas pode ser conseguida através da medição de continuidade entre pontos da matriz, ou mesmo pelo monitoramento da corrente consumida na medida em que novas chaves vão sendo acionadas.

A quantidade de componentes disponíveis na matriz de interconexão interfere diretamente no grau de flexibilidade dos circuitos passíveis de serem montados e analisados. A introdução de componentes programáveis tais como décadas de resistores controladas pelo sistema. Dessa forma, não se amplia o número de componentes efetivamente disponível, mas a gama de possibilidades e com elas novas oportunidades de aprendizado podem ser introduzidas.

A introdução de uma interface com o usuário capaz de permitir que a captura esquemática seja realizada diretamente através do navegador, assegurando dessa forma circuitos que contenham apenas os componentes disponíveis na matriz pode tornar o sistema mais amigável ao usuário, aumentando seu nível de aceitação pelos estudantes, dada a maior facilidade e, eventualmente, velocidade de operação do sistema.

Políticas para o gerenciamento de múltiplos acessos ao laboratório também constituem campo que permite novos desenvolvimentos. Dentre as várias estratégias possíveis pode-se citar a definição de um intervalo de tempo definido para cada usuário, que pode ser ou não função da existência de outros usuários desejando utilizar os recursos. Outra política possível prevê a implementação de meios que permitam a alocação dos recursos à um determinado usuário durante certo período de tempo. Por fim, a interação do usuário com os recursos também pode ser idealizada de forma assíncrona, onde a descrição do circuito juntamente com as informações destinadas aos equipamentos seriam enviadas ao servidor, que as processaria

conforme a disponibilidade de recursos, enviando os resultados obtidos para o usuário por correio eletrônico, por exemplo. Dessa forma, qualquer interação posterior deveria ser realizada através de novo envio.

Referências Bibliográficas

[1] Wullf,C., Ytterdal,T., Saethre,T.A., Skjevan,A., Fjeldly,T.A., “Next generation lab – A Solution For Remote Characterization Of Analog Integrated Circuits” Fourth IEEE International Caracas Conference on Devices, Circuits and Systems, Aruba, April 17-19, 2002.

[2] Fernandez, Rodrigo O., Borges, Adriana P., Peixoto, Nathalia V. Peres-Lisboa, Mauricio e Ramirez-Fernandez, Francisco J. “Laboratório Virtual Aplicado À Educação A Distância“. SBIE2000 - XI Simpósio Brasileiro de Informática na Educação, Maceió, AL. Publicado em Novembro de 2000.

[3] Ferrero, A., Piuri, V., “A Simulation Tool for Virtual Laboratory Experiments in a WWW Environment” IEEE Transactions on Instrumentation and Measurements, Vol. 48, No. 3, June 1999.

[4] Miele D. A., Potsaid B., Wen J.T., “An Internet-based Remote Laboratory for Control Education”, Proceedings of the American Control Conference, Arlington, VA June 25-57, 2001.

[5] Hodge,H., Hinton,H.S., Lightner,M., “Virtual Circuit Laboratory” 30th ASEE/IEEE Frontiers in Education Conference, Kansas City, MO, October 18-21, 2000.

[6] Strandman, J.O., Berntzen, R., Fjeldly, T.A., Ytterdal, T.,Shur,M.S., “LAB-on-WEB: Performing Device Characterization Via Internet Using Modern Web Technology” ” Fourth IEEE International Caracas Conference on Devices, Circuits and Systems, Aruba, April 17-19, 2002.

[7] Fjeldly,T.A., Shur, M. S., Shen, H., Ytterdal,T., “Automated Internet Measurement Laboratory (AIM-Lab) for Engineering Education” 29th ASEE/IEEE Frontiers in Education Conference, San Ruan, Puerto Rico, November 10-13, 1990.

[8] Gustavsson, I., “Remote Laboratory Experiments in Electrical Engineering Education” Fourth IEEE International Caracas Conference on Devices, Circuits and Systems, Aruba, April 17-19, 2002.

[9] Billaud,M. Zimmer, T., Geoffroy,D., Danto,Y., “Real Measures, Virtual Instruments” Fourth IEEE International Caracas Conference on Devices, Circuits and Systems,

Aruba, April 17-19, 2002.

[10] Gomez,F.J., Garrido,J., Martinez,J., “Programming Reconfigurable Hardware Throught Internet” (obtido diretamente do *site* do autor)

[11] Fowler,E.R., Hudson,W.B. “Distance Education from the Faculty Perspective” IEEE Frontiers in Education Conference, 1994.

[12] Tanner, R., Asumadu, J.A., Belter, J., Fitzmaurice, J., Kelly, M., Koh, S.C., Ogunleyeh, H., “Remote Wiring and Measurement Lab ” 31th ASEE/IEEE Frontiers in Education Conference, Reno, NV, October 10-13, 2001

[13] Gómez, F. J., Cevera, M., Matínez, J., “A World Wide Web Based Architecture for the Implementation of a Virtual Laboratory ” (obitdo diretamente junto ao *site* do autor.)

[14] Shen, H., Xu, Z., Dalager, B., Kristiansen, V., Strom,O., Shur, M. S., “Conducting Laboratory Experiments over tthe Internet”, IEEE Transactions on Education, VOL. 42, No. 3, August, 1999.

[15] Flanagan, D., “JAVA in a Nutshell”, 3rd Edition, O’Reilly, 1999.

[16] “LabVIEW User’s Manual”, National Instruments, 1998.

[17] “G Programming Reference Manual”, National Instruments, 1998.

[18] T. Berners-Lee, R. Fielding, H. Frystyk, “RFC1945 - Hypertext Transfer Protocol -- HTTP/1.0”, May 1996

[19] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, T. Berners-Lee, “RFC2068 - Hypertext Transfer Protocol -- HTTP/1.1”, January 1997.

[20] J. Franks, P. Hallam-Baker, J. Hostetler, P. Leach, A. Luotonen, E. Sink, L. Stewart, “RFC2069 - An Extension to HTTP: Digest Access Authentication”, January 1997.

[21] E. Nebel, L. Masinter, “Form-based File Upload in HTML”, November 1995.

[22] ANSI/IEEE 488.1 – 1987, IEEE Standard Digital Interfaec for Programmable Instrumentation.

[23] ANSI/IEEE 488.2 – 1992, IEEE Standard Codes, Formats, Protocols and Common Commands.

[24] K. Dembowski, “PC-gesteuerte Mebtechnik: der PC in der Meb-, Steuer- und Regelungstechnik”, Markt&Technik Editora, 1993.

[25] ___,“Measurement and Automation Catalogue - 2001”, páginas 728 à 735, National

Instruments, 2001.

[26] http://www.ivifoundation.org/

[27] E. K. F. Lee and G.Gulak, “A CMOS Field-Programmable Analog Array”, IEEE Journal of Solid-State Circuits, Vol. 26, No. 12, December 1991.

[28] Fan H., Liu J., Wu Y., “Combinatorial Routing Analysis and Design of Universal Switch Blocks”, IEEE, 2001.

[29] Fan H., Liu J., Wu Y., “On Optimum Switch Box Designs for 2-D FPGAs”, IEEE, 2001.

[30] http://www.metaltex.com.br

[31] Gondran M., Minoux M., “Graphs and Algorithms”, John Wiley & Sons, 1984.

[32] Kreyszig E., “Advanced Engineering Mathematics”, John Wiley & Sons,1993.

Apêndice A - Configuração do DSN – Data Source Name.

Para permitir o acesso de uma aplicação a um banco de dados de forma que o gerenciador de banco de dados utilizado possa ser abstraído é necessária a especificação do chamado *Data Source Name*, realizada através do *ODBC Manager* acessível através do painel de controle.

Open Database Connectivity (ODBC) é uma interface da Microsoft que permite que aplicações possam ter acesso a dados provenientes de uma grande variedade de sistemas de gerenciamento de banco de dados, conferindo independência entre a aplicação e o sistema de gerenciamento adotado (ACCESS, MySQL, etc). É necessária entretanto nesse cenário a presença de um *driver* ODBC apropriado, conforme ilustra a figura abaixo.

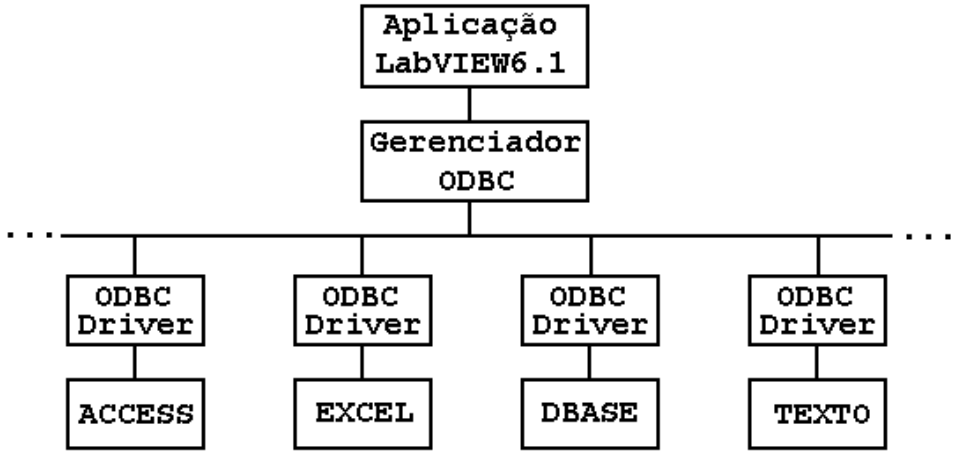


Figura 40 - Estrutura ODBC.

A correta configuração é alcançada pelas etapas apresentadas na página seguinte.

Figura 41 - Ícone de acesso ao ODBC.

Através do painel de controle é possível ter acesso ao Gerenciador ODBC.

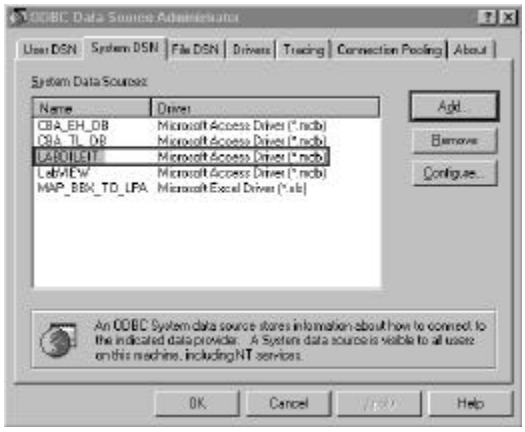


Figura 42 - Configuração do ODBC.

Através desse gerenciador é possível a configuração de diferentes origens de dados e que, de acordo com as folha em que forem adicionadas terão diferentes visibilidades. A origem de dados configurada para o laboratório foi adicionada à folha *System DSN* para que possa estar acessível a qualquer usuário do servidor.

O *driver* ODBC utilizado, conforme indica a ilustração ao lado, permite o acesso à um banco de dados no formato *.mdb, ou seja, estabelecido em ACCESS.

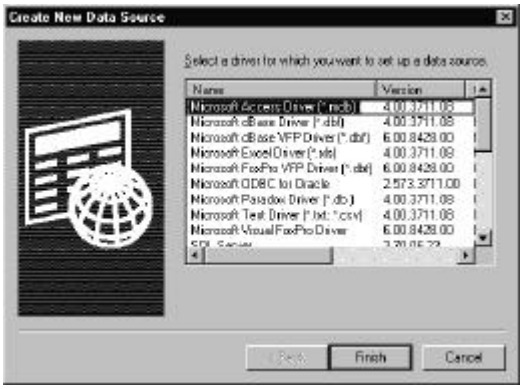


Figura 43 - Seleção do driver ODBC utilizado.

Após a criação de uma nova fonte de dados é hora de configurarmos qual driver ODBC deverá ser utilizado. Uma posterior migração para outro sistema gerenciador significa a alteração do driver empregado.

Figura 44 - Configuração do driver ODBC selecionado.

A tela de configuração final está estreitamente ligada ao tipo de driver utilizado. A figura ao lado ilustra a configuração de uma fonte de dados ligada à um banco de dados em ACCESS.

Apêndice B - Esquemáticos e Layout da Matriz de Interconexão

Neste apêndice são apresentados os esquemáticos completos do projeto da matriz de interconexão, juntamente com as vistas das placas de circuito impresso utilizadas. O objetivo desse apêndice é servir como referência para o caso de futuras intervenções no hardware se tornarem necessárias.

A figura 43 contém o esquemático do registrador de deslocamento de 60 bits implementado através de oito circuitos integrados 74164 — *8-Bit Parallel-Out Serial Shift Register*. Ao lado de cada um dos circuitos integrados, um capacitor cerâmico de 100nF foi alocado.

A figura 44, em sua parte inferior, contém a fonte de alimentação que é formada apenas por um circuito integrado regulador de tensão 7805, capacitores, e um LED indicador de operação.

Na parte superior da figura 44 é apresentado o circuito responsável pelo endereçamento de cada uma das placas que compõem a matriz. Um circuito integrado comparador de 4 bits — 7485 — é utilizado para a comparação entre o endereço programado na placa e o endereço presente no barramento. Considerando o elevado número de cargas alimentadas pelas linhas de RESET e CLOCK foi intercalado em cada uma delas um *buffer* formado por um transistor extraído do ULN2003.

As figuras 45 a 49 trazem os esquemáticos de cada um dos 10 hipernós acondicionados em cada uma das placas. Em cada um desses hipernós encontram-se 6 relés SH1NAC-5V cada um deles acionado por um dos transistores presentes do circuito integrado ULN2003 — *Darlington Transistor Array*.

As figuras 50 e 51 ilustram as faces da placa de circuito impresso projetada. O roteamento foi realizado de forma manual nas seções mais críticas e de forma automática em outras partes.

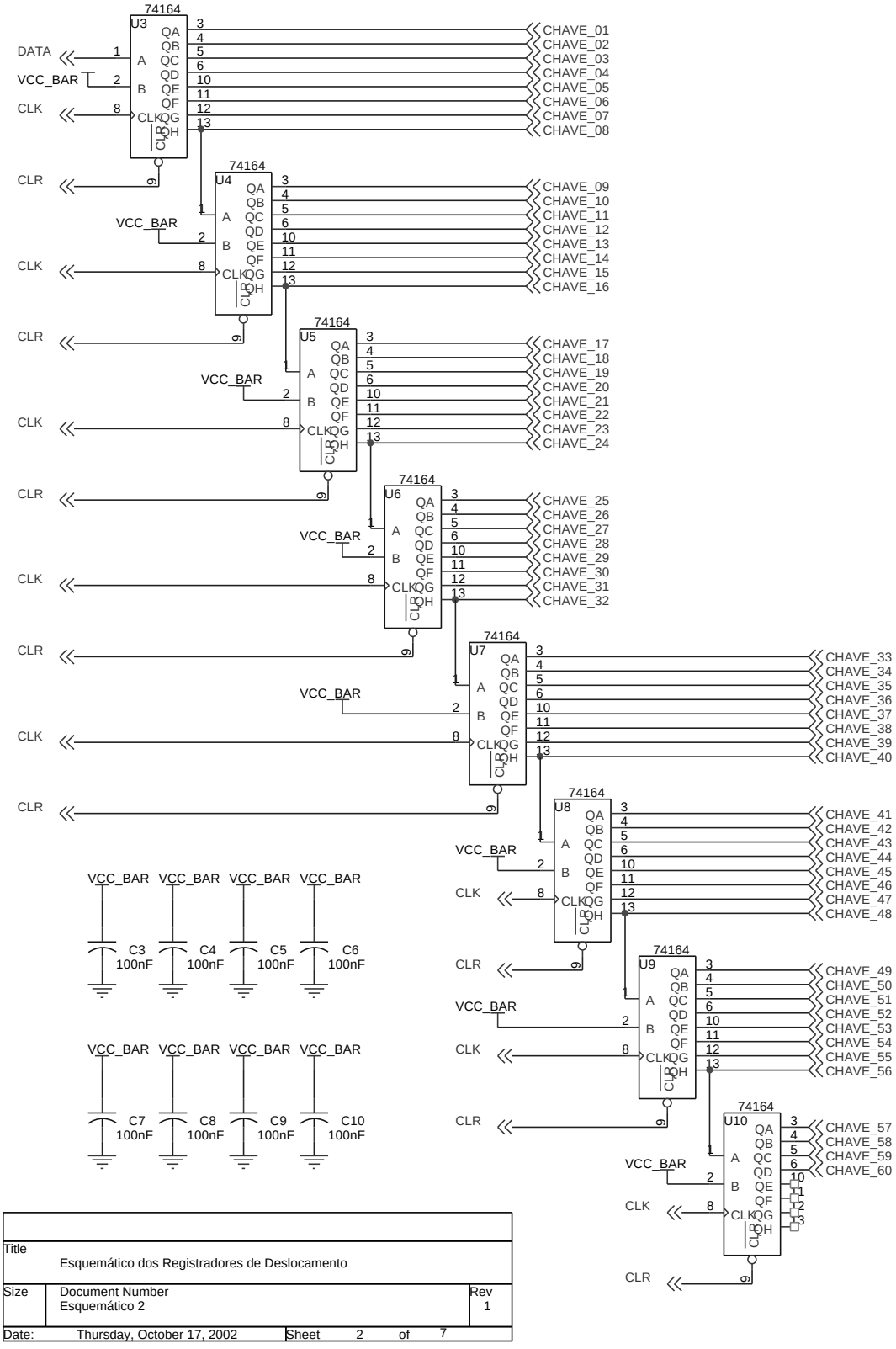
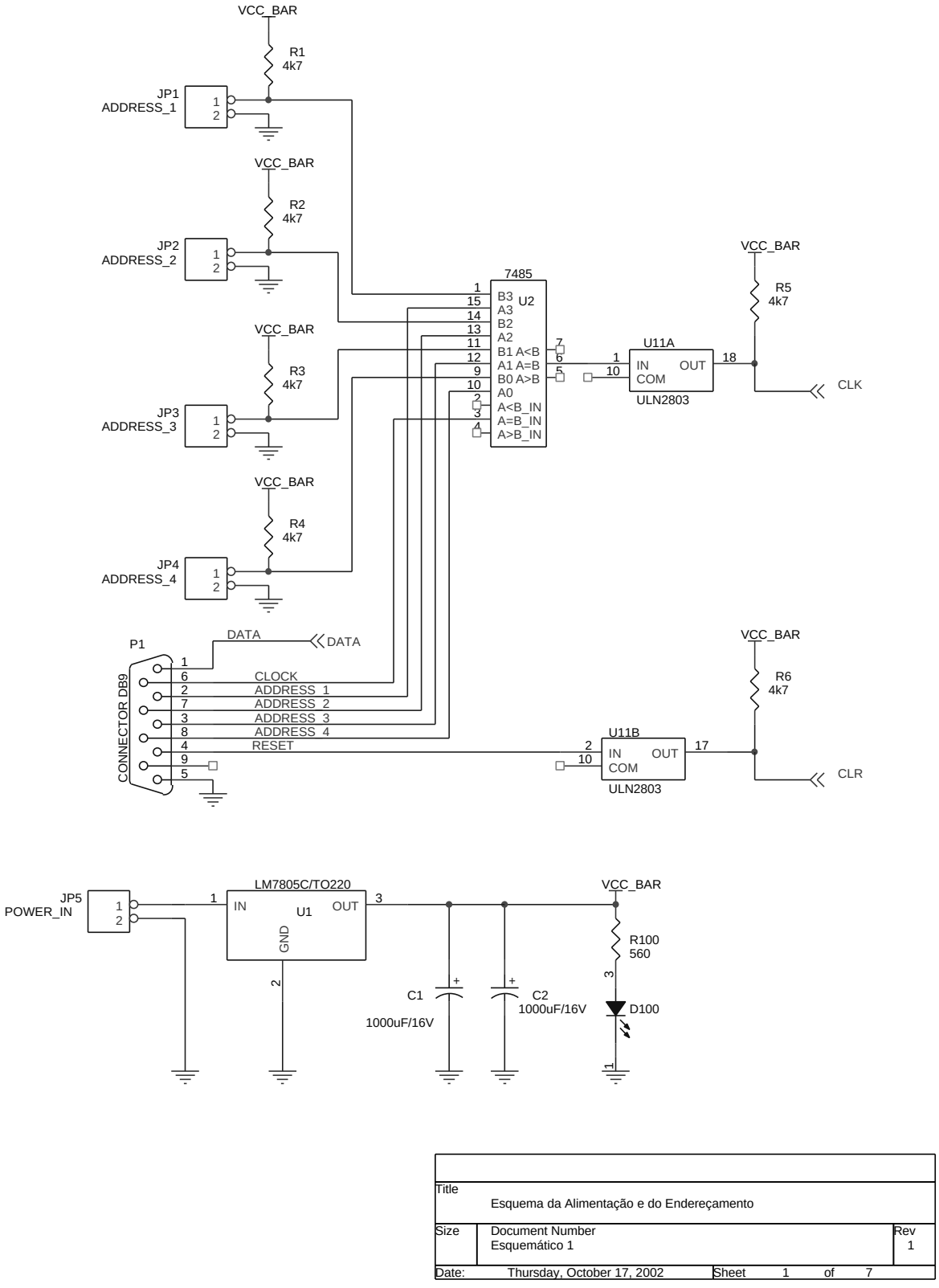


Figura 45 - Esquemático do registrador de deslocamento de 60 bits.



Title: Esquema da Alimentação e do Endereçamento		
Size	Document Number: Esquemático 1	Rev: 1
Date: Thursday, October 17, 2002	Sheet 1 of 7	

Figura 46 - Esquemático do decodificador de endereços e fonte de alimentação.

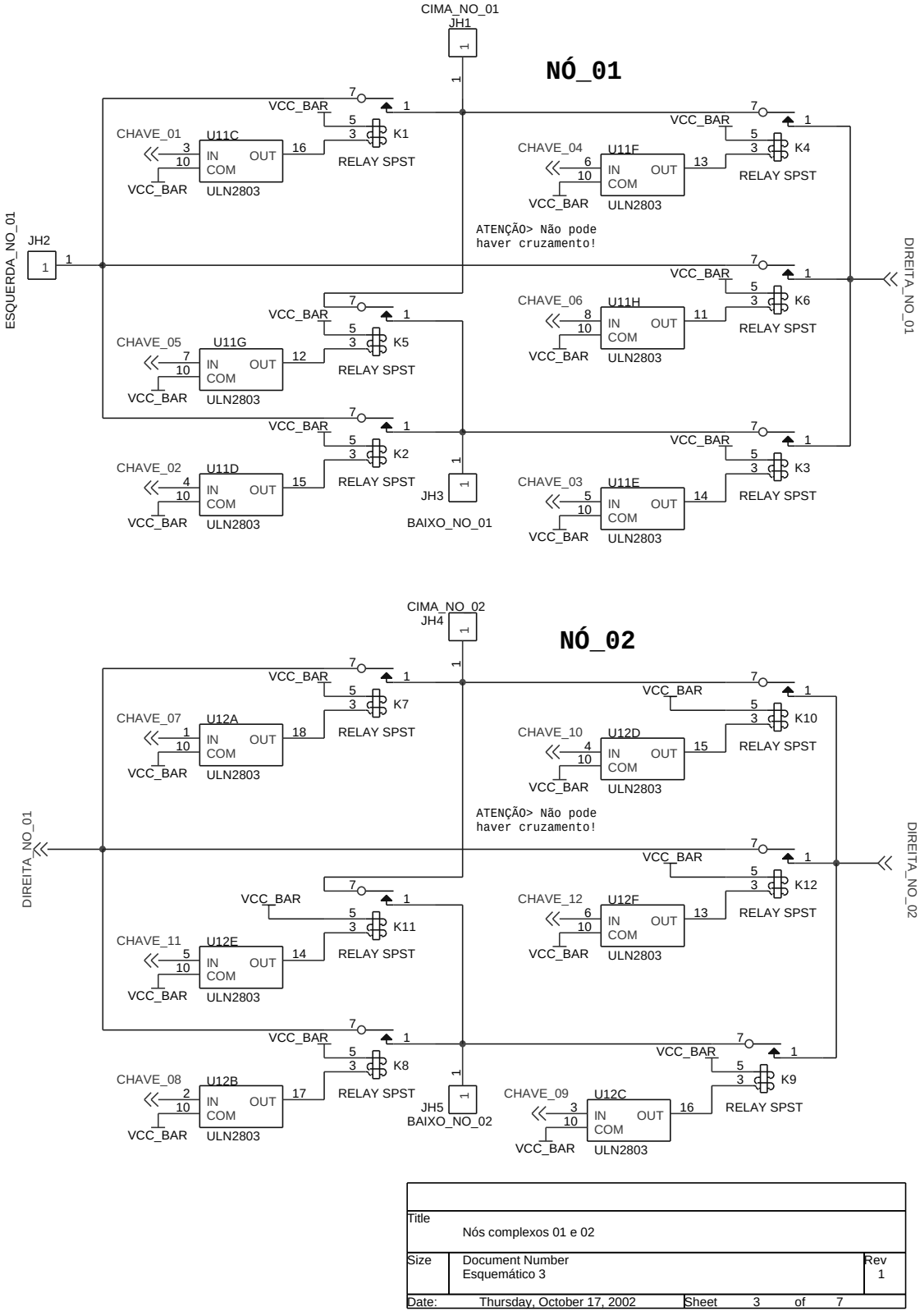


Figura 47 - Esquemáticos dos hipernós 1 e 2.

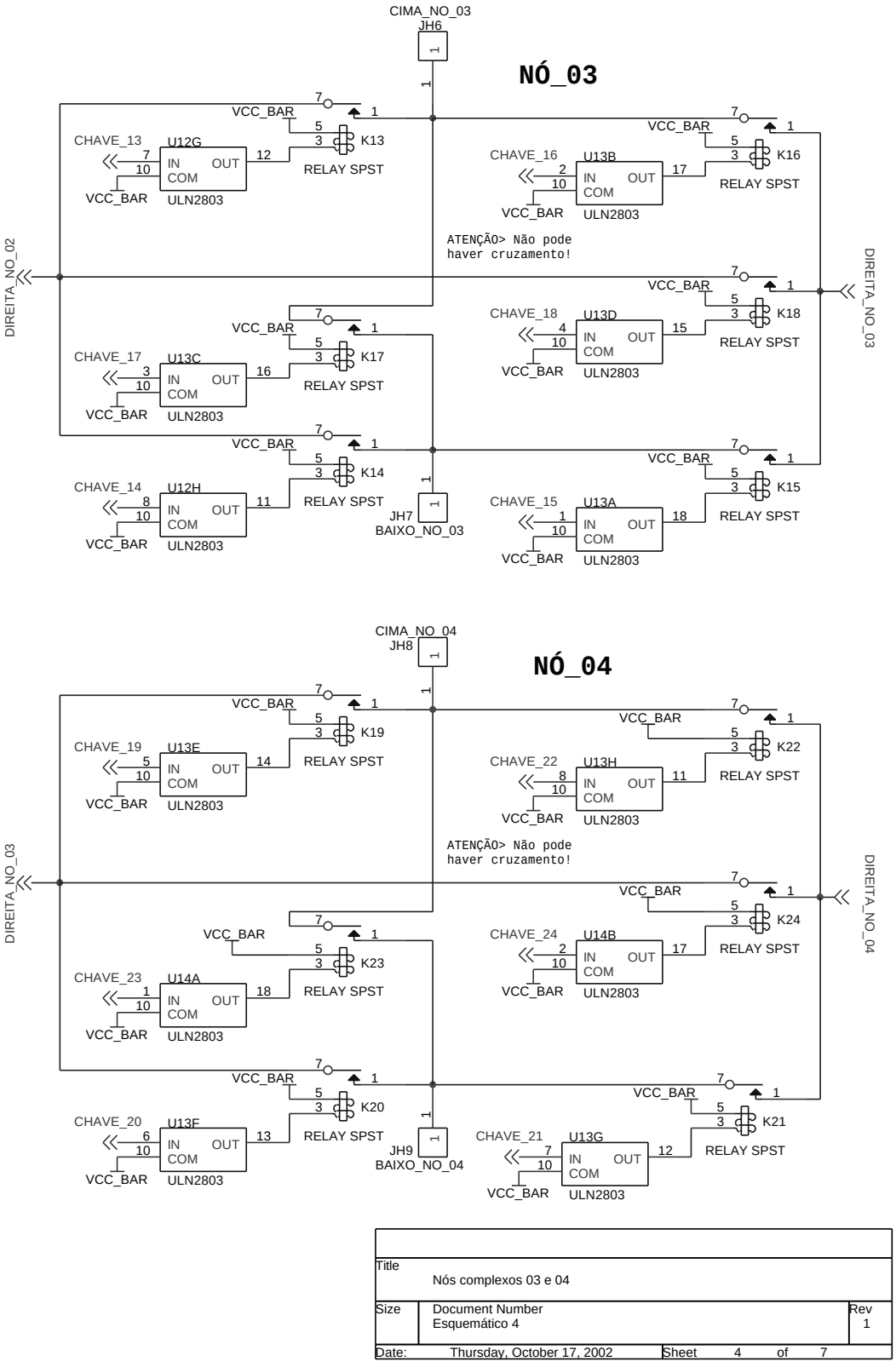


Figura 48 - Esquemáticos dos hipernós 3 e 4.

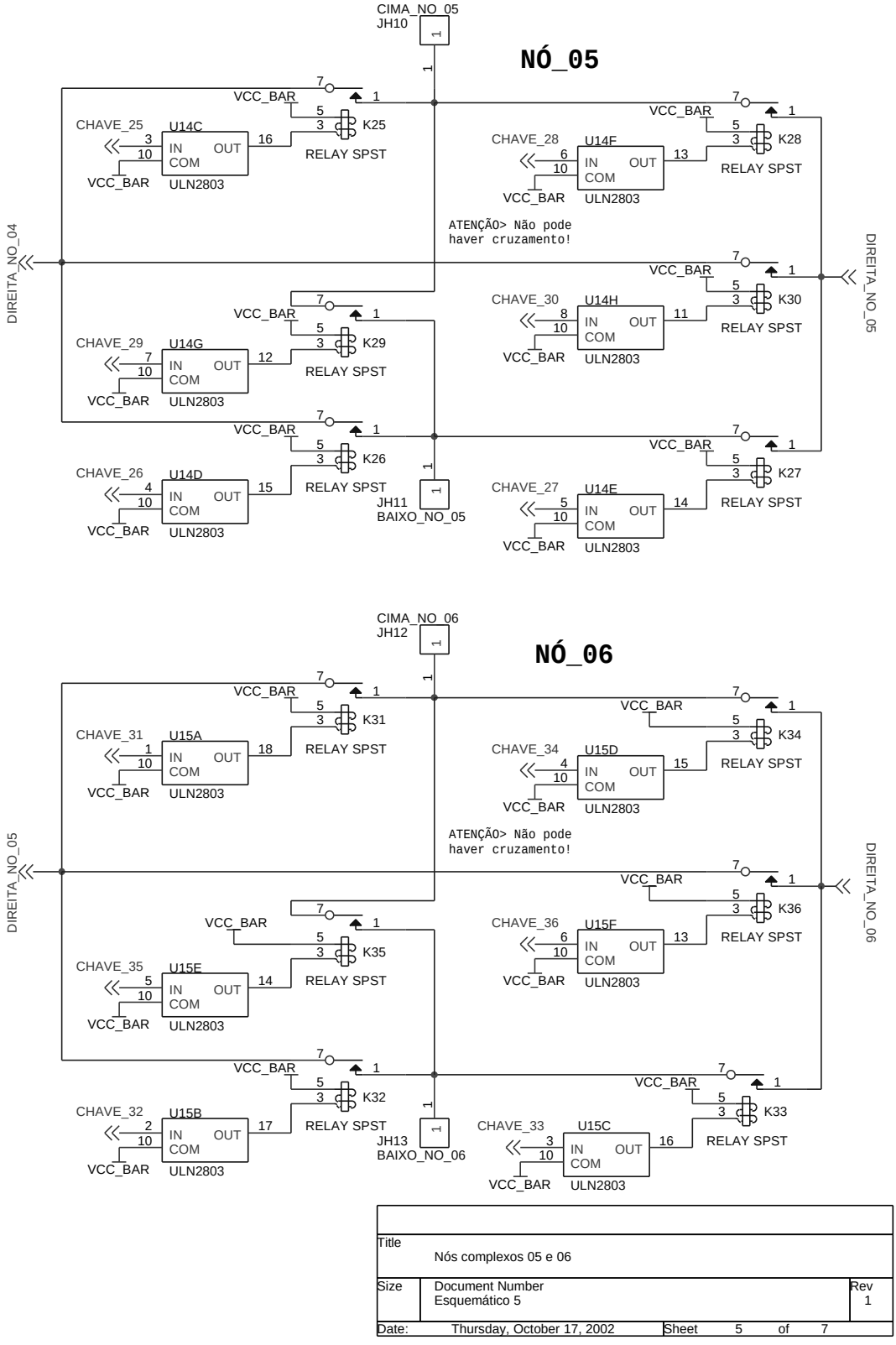


Figura 49 - Esquemáticos dos hipernós 5 e 6.

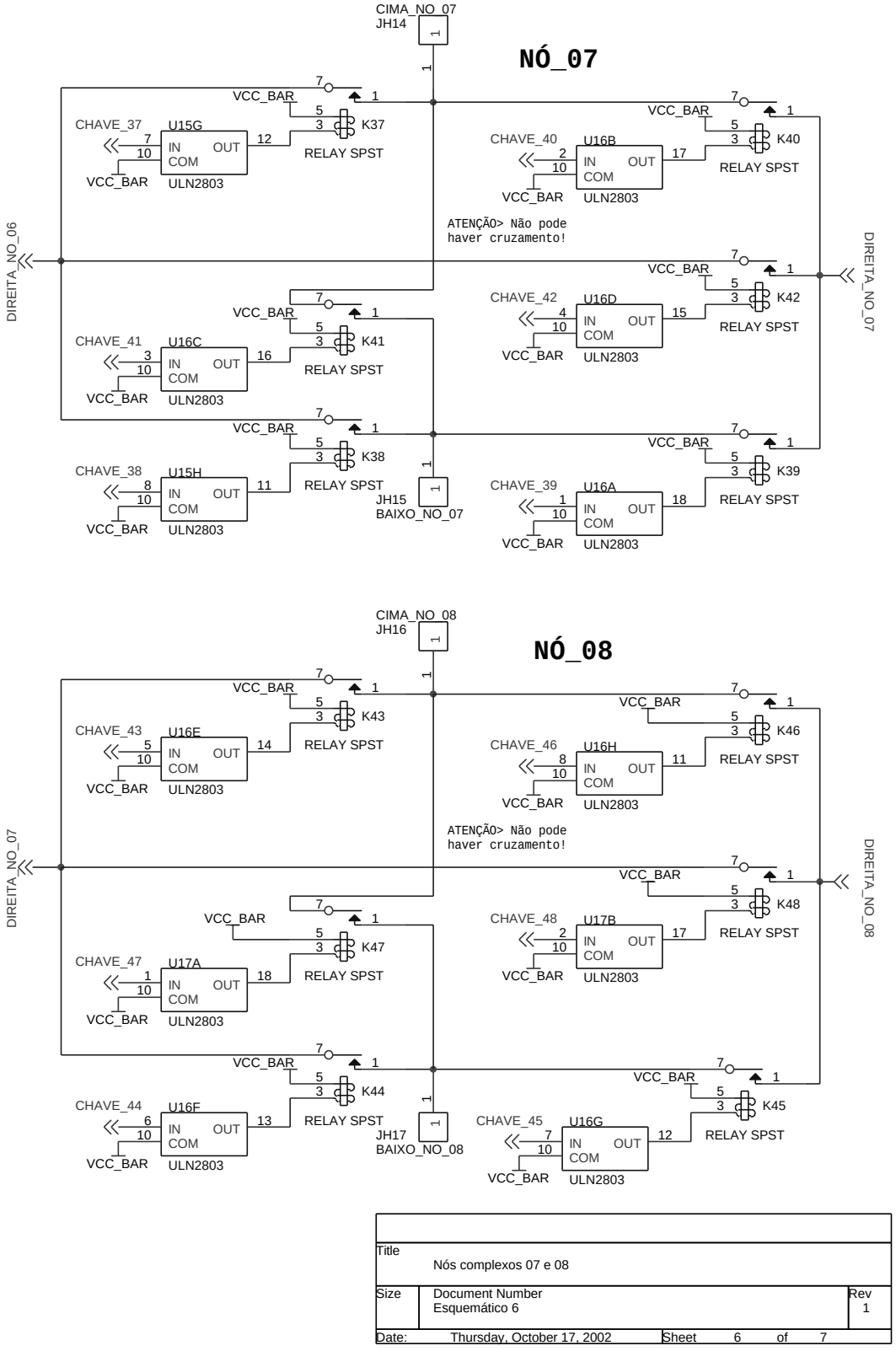


Figura 50 - Esquemáticos dos hipernós 7 e 8.

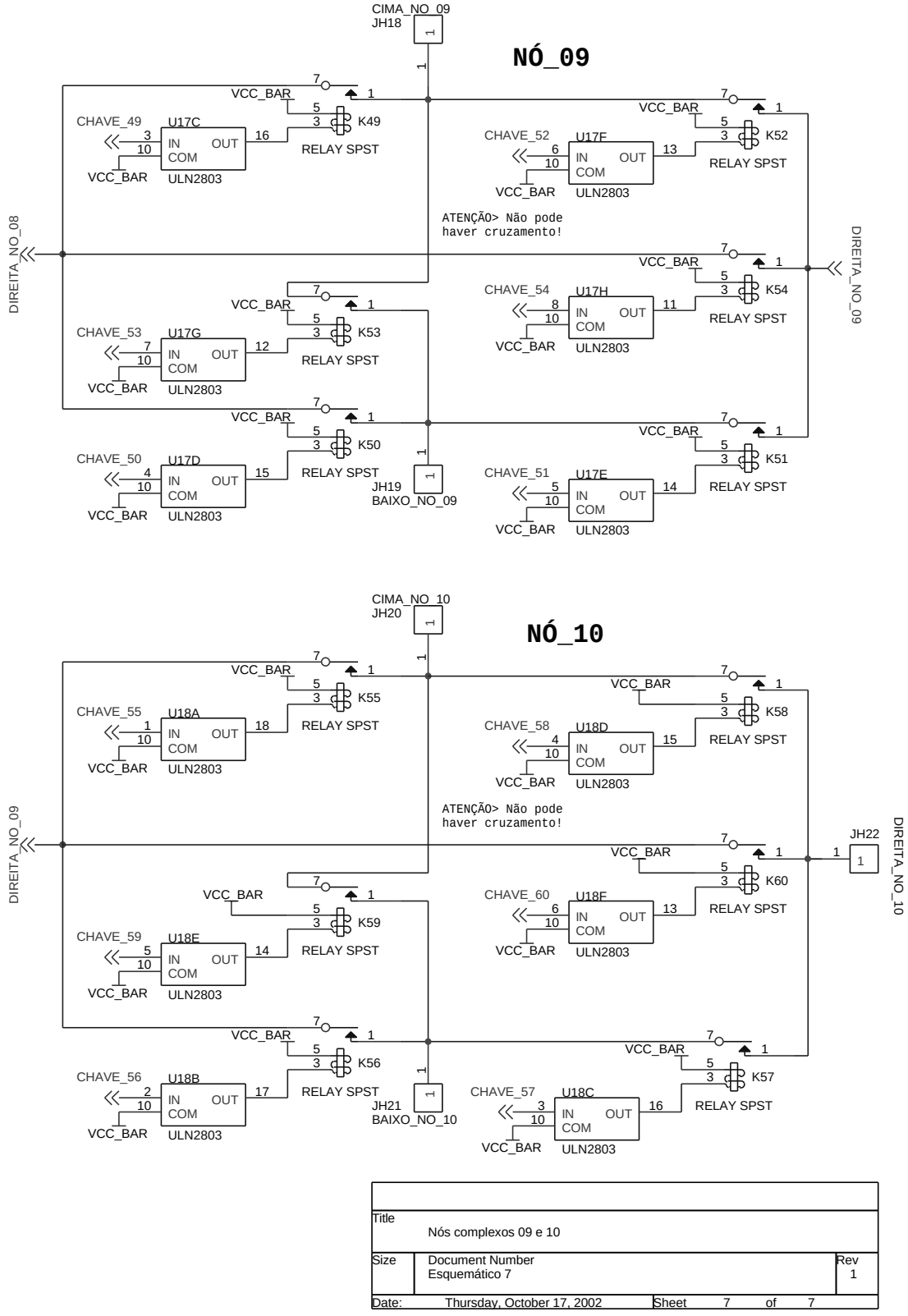


Figura 51 - Esquemáticos dos hipernós 9 e 10.

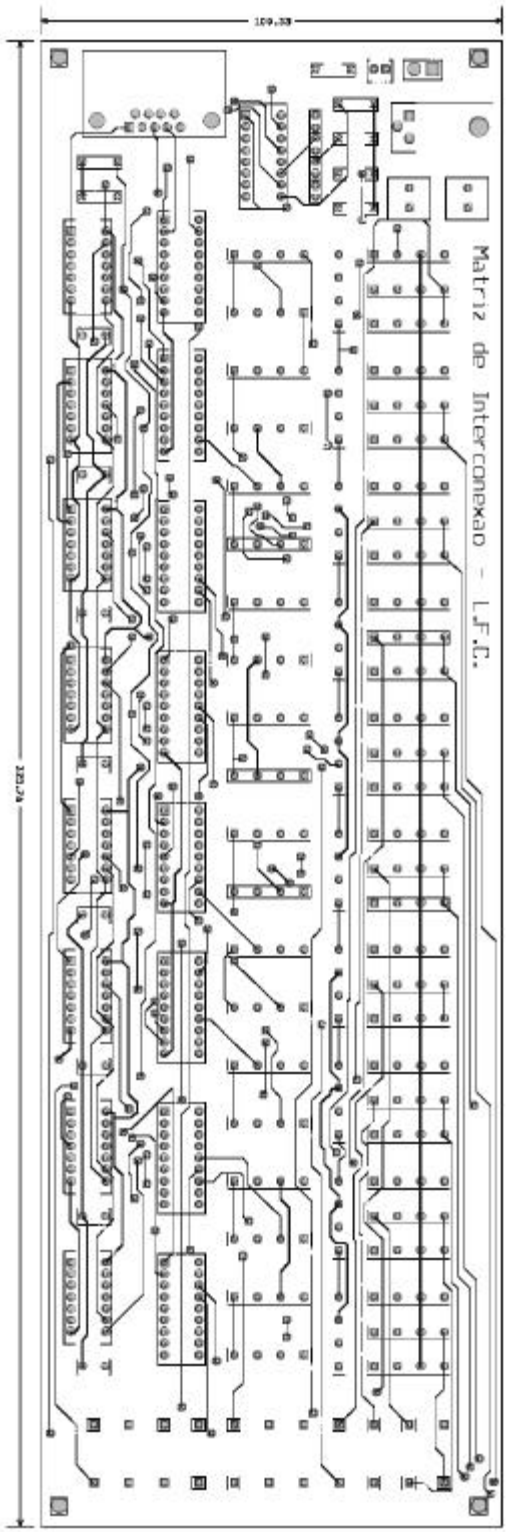


Figura 52 - Vista superior (lado dos componentes)

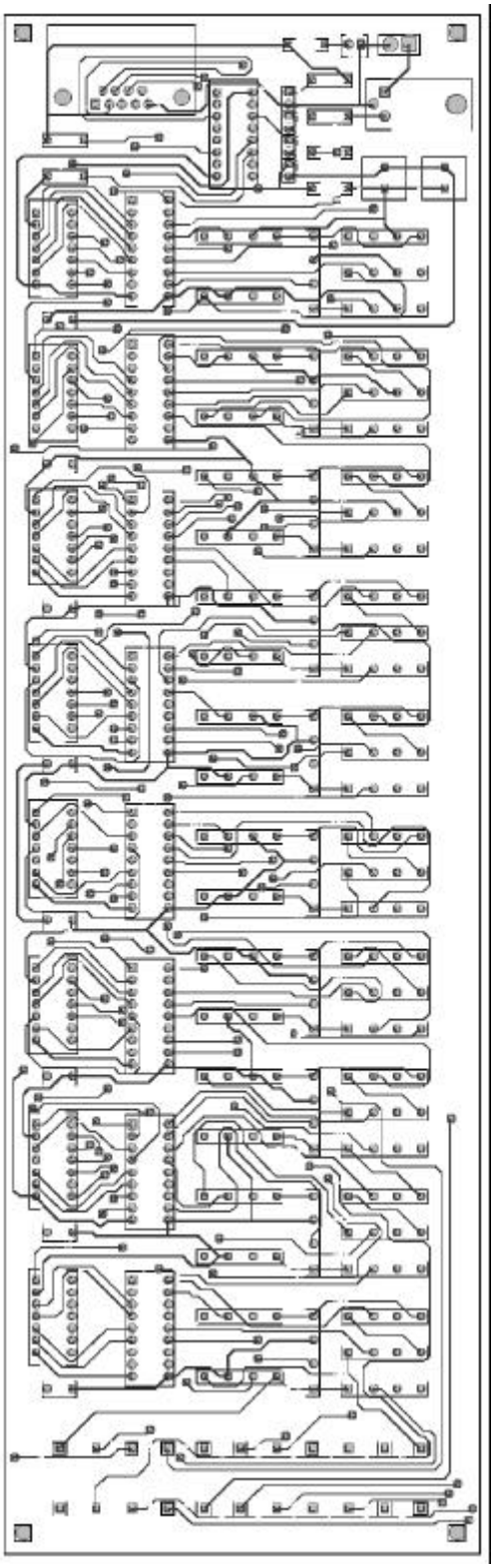


Figura 53 - Vista inferior (lado das soldagens)